

Evolution of Data Warehousing and Data Mining

Peter Parapatics
Matrikelnummer: 0225859
Studienkennzahl: 534

Seminar (mit Bachelorarbeit) 188.174
SS 2007

Advisor: Dipl.-Ing. Dr. Alexander Schatten

24. Juni 2007

Erklärung: Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus den Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Wien, 24.06.2007

Inhaltsverzeichnis

1 Abstract	5
2 Introduction	5
3 Beginnings of Data Management	7
3.1 First computers - Data management characteristics	7
3.1.1 Calculation intensive	7
3.1.2 Data accessibility	7
3.1.3 Batch processing	7
3.1.4 Redundancy	8
3.1.5 Centralized	8
3.2 Evolution of data management	8
3.2.1 Evolution of storage devices	9
3.2.2 Software Evolution	9
3.2.3 Relational Databases	9
4 Data Warehousing - From operational to analytical data	11
4.1 Complexity issues in data warehousing	11
4.2 Content characteristics of data warehouses	12
4.3 Types of data in a data warehouse	13
4.4 Data organization and operations in data warehouses	14
4.5 Persistence strategies in data warehouses	15
4.6 Data extraction for data warehouses	16
4.7 Types of data warehouses	16
4.7.1 Enterprise-wide data warehouse	16
4.7.2 Data marts	16
4.7.3 Virtual data warehouse	17
4.8 Impacts of data warehousing	17
4.8.1 Data mining	17
4.9 Evolution of data warehousing and data mining	18
5 Evolution of the Internet	19
5.1 The beginnings - ARPANET	19
5.1.1 Birth of the Internet	19
5.1.2 First applications	19
5.2 WEB	19
5.2.1 Architecture of the WEB	20
5.2.2 Consequences for data mining	20
5.3 Evolution from WEB to WEB 2.0	20
5.3.1 Architecture of participation	21
5.3.2 Social Networking	21
5.3.3 Data driven applications	21

5.3.4	Context independent data	22
5.3.5	Service Orientated Architecture	22
5.4	Semantic Web	23
5.4.1	Architecture of the Semantic Web	24
6	Recent trends in data warehousing and data mining	26
6.1	EII - Enterprise Information Integration	26
6.1.1	Differences between EII and EAI	26
6.1.2	Advantages of EII	27
6.1.3	Disadvantages of EII	27
6.1.4	Advantages and Disadvantages - Implications for EII	28
6.1.5	EII and Semantic Web	28
6.2	Web mining - The World Wide Web as data source	30
6.2.1	Subcategories of Web Mining	30
6.2.2	Web mining and Semantic Web	31
6.3	Multimedia data mining	33
6.3.1	Text mining	33
6.3.2	Audio mining	33
6.3.3	Image and video mining	33
6.4	Data streams	34
6.4.1	Streaming applications	35
6.4.2	Issues in data stream mining	36
6.4.3	Stream classification	37
6.4.4	Real time stream processing	37
6.4.5	Stream processing algorithms	38
6.4.6	Event Driven Architecture	40
7	Summary	41

1 Abstract

With the evolution of hardware and software and with the possibility of transferring large amounts of data the number of available and interesting data sources has constantly increased. As a result computer systems have evolved from simple calculation systems to complex data management systems. The need for analysing these large amounts of data to extract knowledge has increased.

This thesis describes characteristics of the first computers and analyses the evolution towards complex data warehousing and data mining systems. Technologies influencing and contributing to this evolution are described. A focus is set on the evolution of the Internet and especially WEB 2.0 characteristic important for data mining and data warehousing.

The main focus is set on recent data mining and data warehousing trends. Therefore Enterprise Information Integration, Web Mining, Multimedia Data Mining and Data stream mining are analysed in detail.

2 Introduction

From early history on men were interested in persisting information and data for various purposes. Cave paintings show every day life scenes and even before paper was invented clay plates existed for documentation of trading. So during the entire history of mankind data management has been an important field. People soon found out that data and information is great source of power. Since in many fields the one with more information is in a superior position, collecting relevant information for decision support has always been an important field in data management. An example that goes through the entire history up to now is military tactics in warfare. Knowing more than the enemy has been the key factor for success in many battles. Knowing the strength and even more important the strategy of an enemy allowed developing counter strategies. Therefore generals lost battles due to the lack of information, although they had the stronger army. Another important field has always been predicting the “future” based on the acquired data. This requires analysing data on a higher level of abstraction to extract hidden information. Generals needed to draw conclusions about the “global” strategy of an enemy instead of only focusing on one specific battle. Information like the position of an enemy and possible intends must be integrated in order to predict further actions. This thesis focuses on knowledge extraction and decision support in computer-aided data management. Therefore the evolution from the first computer systems towards complex data warehousing and data mining systems is examined.

Chapter three describes the evolution from punch card systems to computers where data is accessible in real-time. It depicts the characteristics and shortcomings of the first computers and the purposes for which these computers were used. Furthermore the way data was administrated and technologies crucial for the evolution of data management are analysed and illustrated by an

example of an Austrian insurance company¹. Consequences for data management and especially data analysis are examined.

Chapter four deals with the separation of computer systems into operational and analytical systems. It describes reasons for this separation and differences between analytical and operational systems. Therefore requirements for data warehouses and operational systems are compared. The focus lies on the evolution of data warehouses and how data is organized in data warehouses. Three fundamental types of data warehouses with respect to size and organization are described. The data extraction process for data warehouses is described and different kinds of data available in data warehouses are explained. Furthermore the beginnings of data mining based on data warehoused are examined.

Chapter five describes the evolution of the Internet since many advances in data mining and data warehousing are based on increased connectivity of computer systems. A short historical overview is given, followed by the basic concepts of the Web 2.0 important for data mining. It is described how data is organized, which additional data sources are available and some software architectures contributing to Web 2.0. The architecture and goals of the Semantic Web, as an upcoming trend that might contribute to data mining, is described finally.

Chapter six analyses current trends in data mining and data warehousing, resulting from the evolution of the Internet and increased processing and storage capacities of computers. The Web itself is a large data source that can be mined web mining is one important topic. Especially how the Semantic Web can contribute to web mining is analysed. Another important topic is data integration over various distributed data sources. Enterprise Information Integration (EII) is described as one technique for this purpose. The difference between Enterprise Application Integration and Enterprise Information Integration is explained and advantages and disadvantages of EII are analysed. Finally it is analysed how the Semantic Web might contribute EII. Since an increased amount of multimedia data is available text mining, audio mining and video mining are briefly explained. A focus is set on data stream mining, since much data is no longer available from “static” data sources like relational databases but arrives continuously as stream. Examples of streaming applications are given as an introduction to explain why stream mining is an upcoming field. Based on these examples issues and requirements in data stream management are analysed. The most important issue for data stream mining is real-time capability. Due to the large amount of data created by data streams most streams must either be mined in real-time or relevant data must at least be extracted from streams in real-time in order to reduce storage requirements in data warehouses.

¹The example is based on information I received from my father, Helmut Parapatits, who had been working in this company during the 1960's.

3 Beginnings of Data Management

The evolution of electronic data management did not start as one would expect with the invention and construction of the first computers. The first computers did not show several important characteristics of modern data management systems. Therefore it is very difficult to name a specific point in time when computer systems evolved from merely calculation to data management systems. To see the difference to modern data management systems it is helpful to examine the characteristics of the first computer systems.

3.1 First computers - Data management characteristics

3.1.1 Calculation intensive

The first computers were constructed to ease difficult mathematical calculations. To me it seems, that originally there was no intent to use computers for data management. One of the first computers was Colossus created and used by the British government to break German codes during World War II. The first programmable computer was ENIAC which was built in 1946[4].

For many business companies computers became cost efficient only in the 1960's. Due to the lack of data storage devices supporting real-time read and write operations computers were still used for calculations only. But since these computers were programmable it was not feasible to enter data to be calculated by hand. So in fact although computers were not intended for data management a computer readable storage media was needed.

3.1.2 Data accessibility

The first widely adopted storage media were punch cards. Since these punch cards were stored in an external archive data was not immediately accessible.

An Austrian insurance company for example had all their contracts with customers on punch cards. A common query like "all contracts from customer A" was not executable using a computer terminal. Such queries were performed by going to the archive and looking up the appropriate punch card. Punch cards were sorted by contract numbers. In order to find all customer's contracts indexes existed in the form of human readable index cards².

3.1.3 Batch processing

Since data was stored on external sources calculations as well as updates had to be done as batch process.

If data had to be changed it was necessary to replace several punch cards with punch cards containing new data. Since it was not feasible to perform updates, changes were collected and synchronized with existing data in fixed time intervals. Updating existing data was a very time

²The example is based on information I received from my father, Helmut Parapaties, who has been working in this company during the 1960's.

consuming process although computers were able to perform such synchronizations mostly automatically. In order to change a punch card a new punch card had to be printed containing the modified data and an identifier which data record needs to be changed. Computers were able to evaluate such a punch card and synchronize it with the existing punch card by creating a new punch card.

In an Austrian insurance company if a contract had to be changed a punch card was punched containing a change code, the updated data and the contract number. The contract number served as a unique identifier for a contract while the change code indicated which data record needs to be updated. Once a week these punch cards were merged with existing data. The human readable index cards specifying where to lookup customer's contracts were printed automatically. Old punch cards were kept to maintain a change history³.

For most modern computer systems batch updates would be a serious limitation since data is not always up to date. But since computers and the stored data were mainly used for calculations which were done in batch processing as well punch card based data management was a great improvement over non computer based data management.

3.1.4 Redundancy

With punch cards, redundancy was often not avoidable. Punch cards were ordered by a unique identifier. Let's stick to the insurance example with the contract number serving as unique identifier. Every contract had at least one customer. For calculation purposes it was necessary to store customers' data redundantly on each contract punch card. If an update in a customer's address occurred several contract punch cards of this customer had to be updated³.

3.1.5 Centralized

Due to the lack of interconnectivity and high costs of hardware, computer systems were centralized systems designed for one specific purpose. Until computer networks started to evolve and personal computers started to replace mainframes there was no communication with other computer systems for electronic data exchange. No uniform data representation was available. Data on punch cards for example had a proprietary format, so data received from different companies could not be delivered on punch cards. When data was received new punch cards had to be punched containing data in proprietary format.

3.2 Evolution of data management

Data storage for calculations only was a serious limitation for data management. The evolution of data management went hand in hand with the increasing capacities and the support for real-time read and write operations of storage devices. The core technology for many advances in data management are hard disks. This can be seen especially when analysing the evolution of

³The example is based on information I received from my father, Helmut Parapatits, who has been working in this company during the 1960's.

hard discs compared to relational database systems which have become are a core technology for data management.

3.2.1 Evolution of storage devices

It was in the 1950's when IBM first used magnetic tapes to store data. Magnetic tapes and later magnetic drums were capable of storing higher amounts of data and therefore started replacing punch cards. But in fact magnetic tapes had serious draw backs. Data is written to and retrieved from magnetic tapes and drums sequentially which implies that without using special indexes, searching for a data record can result in winding the whole tape until the record is found. Therefore magnetic tapes did not replace punch cards completely. Many companies continued using punch cards as storage media for operational data while using magnetic tapes for backup. Those companies who switched to tapes still managed their data in batch processing mode. It was the hard disk that in fact revolutionized data management. The possibility of random access was the key towards real-time data processing and management. Details on hard disk evolution can be found in [5].

3.2.2 Software Evolution

The evolution of storage media triggered the needed to manage data on these devices.

Along with the first hard disks conceptual work on software started to separate data representation from specific hardware. In the early 1960's the term "database" emerged to capture the sense that the information stored within a computer could be conceptualized, structured, and manipulated independently of the specific machine on which it resided [15]. (Further information concerning early databases can be found in [22].)

This was the first step to separate program code and operational data. Research was at the beginning mostly done by military and US intelligence, but soon adopted for commercial applications [15]. Databases were developed to achieve data independence. By separating databases to a logical and physical schema the first step to separate application code and data was done. This separation of concerns can be seen as a milestone for data management. It is a basic requirement for using data in various contexts and independently of a specific application. Especially the ability to query data sources in real-time opened new possibilities. In punch card archives large amounts of valuable information theoretically existed but were not available due to the mentioned characteristics of punch card systems examined above. Only simple reports, summarizing data were generated. Real-time data access was the first step towards analysis and extraction of patterns and information from large amounts of data.

3.2.3 Relational Databases

In the beginning of the 1970's IBM researcher Edgar F. (Ted) Codd published the paper "A Relational Model of Data for Large Shared Data Banks" introducing the relational data model. The data organization suggested by Codd consists of tables linked by keys identifying specific

data records [14].

With the relational data model Codd tries to

“provide a means of describing data with its natural structure only – that is, without superimposing any additional structure for machine representation poses. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.”

Codd [14]

He lists three dependencies of data that can be avoided with relational databases.

- Ordering Dependency
- Indexing Dependency
- Access Path Dependency

Ordering independence means that an application must not be concerned with the physical ordering of data. This separates the ordering of presentation from ordering of representation. Ordering can still be achieved by writing specific queries but is not imposed by physical storage ordering.

Indexing independence implies that it does not matter whether indexes exist to access data or not. Applications querying a database must receive the same results in either way.

Access Path independence means that the physical location of data is transparent for an application. This is best explained by describing a storage mechanism that does not provide access path independence. Take a tree based file system as an example. If a file is moved an application will cause an error when trying to access it. Relational databases allow querying data records without specifying where the specific data is exactly located.

With relational data bases it became possible to eliminate redundancy. With a normalized database data is separated into non-redundant tables that can be linked.

At the beginning of the 1970's IBM started intensive research on relational databases [2]. The first prototype was accomplished in 1974. In 1979 the first commercial relational database system named Oracle was available using the structured query language SQL. At the beginning different vendors used different query languages but SQL became the de facto standard and was later standardized by ANSI [2].

Relational databases became the core technology for data management. One important reason is the uniform and standardized interface provided through SQL. This implies that applications do not depend on a proprietary file format. One database can therefore be used by different applications for different purposes. While changing the file format of file based

applications very often requires changing the application code, relational databases at least allow extending the logical schema without affecting existing applications. Another important issue is the simplicity of core SQL query language functions. The core functions of relational databases which are “insert”, “update” and “delete” for manipulating and “select” statements for retrieving data can be used with very few syntax learning effort.

Modern relational databases vendors extend this core functions by adding additional vendor specific functions which turns out to be very successful. Specialized databases can be chosen depending on the application domain and the knowledge of the application developers. This flexibility and the increasing performance and robustness of relational databases has established them as the predominant persistence strategy and as a basis for the evolution of data management.

Similarities to punch cards It is noticeable that many characteristics from punch card data management have been adopted in database development. The concept of primary keys has already existed for punch cards. Indexes to speed up queries have exactly the same purpose as human readable index cards. The database management system creates a data structure in order to find entries in the database without parsing all records. So in fact even the technical implementation has the same concept as human readable index cards, therefore suffering from the same drawbacks. Inserting or updating data requires updating the index.

4 Data Warehousing - From operational to analytical data

With the evolution of storage devices and the ever increasing storage capacities, larger amounts of data became manageable. In the beginning of the 1970s researchers from the MIT started to differentiate between operational and analytical systems [23]. From that time on data management started to evolve from storing and using data for daily operational purposes like calculations or customer management towards analysing data in order to use it for decision support. The fundamental idea for data warehousing was born and can be seen as the next major step in data management since relational databases.

Due to limited storage and processing capacities of computer systems it lasted until the 1990's that data warehousing started to make its way into business companies. More and more managers needed to access operational data collected during daily business for decision and planning support. It turned out that operational systems did not provide the means to access most of the data from the operational databases for analytical purposes. This phenomenon was described as “data in jail” [41].

4.1 Complexity issues in data warehousing

Data in operational databases was often restricted to a specific area in an organization but reports and evaluations were needed on a company wide level. A central storage was needed to collect data from various business fields and departments of a company.

An important issue was to be able to analyse data with respect to time - a feature that an operational database does not support since data is always up to date at a specific point in time. Accessing historical data and analysing changes over time is very often not supported by operational databases. Keeping change records in operational databases would have a significant negative impact on the amount of data to be stored and therefore on performance and scalability.

4.2 Content characteristics of data warehouses

It was in 1991 Bill Inmon published the book “Building the data warehouse” describing the main characteristics of a data warehouse and proposing an architecture for developing data warehouses. The key aspect of a data warehouse is that it should be used as a decision support system only and not for operational purposes. Data should be collected from various data sources, for example from operational databases from different areas in a company. The collected data should be saved in a uniform way in order to make it accessible for analytical purposes. So a data warehouse has a much wider scope than doing reports on an operational database of a specific area in a company.

In [23] Michael Haisten describes four major characteristics, originally defined by Bill Inmon, which distinguish data warehouses from operational systems:

- Subject-Oriented
- Integrated
- Time-variant
- Nonvolatile

Subject orientated means that data stored in a data warehouse focuses on the main subjects of a company, for example customers or products. Operational data not needed for decision support must be eliminated. For example the credit card number of a customer which is needed for a purchase transaction is not relevant for analysing customers’ data but the fact that a customer used a credit card for payment might be highly relevant for enterprise wide decision support.

Integrated means that data must be saved in a uniform and consistent way although it might be a collection from several operational databases and different data sources. The end user of a data warehouse like an analyst or manager must not be concerned with the consistency and integrity of the data in the data warehouse.

Time-variant means that data can be accessed for an arbitrary point in time. This implies that changes can be analysed over time. This contrasts operational databases where data is constantly changing.

Nonvolatile in the context of data warehousing means that data once stored in a data warehouse can not be changed any more. This is a major difference to operational data management where data records are continuously changing. Nonvolatile therefore implies that no update but only load and retrieval operations are allowed.

These four characteristics make clear why there is a need for separation of operational and analytical data. The separation was at the beginning also necessary to avoid negative impacts on the performance of operational data sources. Too many analytical tools working on an operational database slowed down operational processes.

4.3 Types of data in a data warehouse

In [1] the different forms of data in a data warehouse defined by Bill Inmon are described:

- Older detail data
- Current detail data
- Lightly summarized data
- Highly summarized data
- Meta data

The separation of detailed, lightly summarized and highly summarized data gives analysts the flexibility to switch between granularities of data.

The separation between older detail data and current detail data positively influences scalability. Older data can be stored on cheaper external storage media while for newer data technologies can be used which provide fast and efficient data access methods.

Like in many other fields of data management meta data continuously gained importance during the evolution of data warehouses. Kimball differentiates between the following types of meta data [30]:

- Source system meta data
- Data staging meta data
- Database management system meta data

Source system meta data is data describing the data sources that are extracted with information like update frequencies of sources, business descriptions of each source or legal limitations on the use of each source . Due to the ever increasing complexity and number of data sources it is very important to have such data available since some sources are very likely to change frequently. Therefore keeping source meta data up to date is important. Software developers need to know how to access relevant data in order to extract it for the data warehouse.

Data staging meta data generally speaking describes what happens to data before it is loaded into the data warehouse. For example how data is stored and transformed before it is transferred to the data warehouse. Which fields are taken from which source? Which data is aggregated? Which information is filtered? Is data being transformed or enhanced by for example expanding abbreviations .

Database management system meta data describes technical details like table definitions, view definitions or stored procedures of the data warehouse which can be used by for example analytical tools to access the warehouse. Furthermore update or security policies of the data warehouse can be specified.

Generally it can be differentiated between technical and business meta data. Business meta data is information on how data was collected, for example if it was preprocessed before it was stored in the data warehouse. Technical meta data provides information about data structures and access methods that can be used to analyse business data.

4.4 Data organization and operations in data warehouses

Due to different requirements and use cases in data warehouses and operational systems the way data is logically organized differs. It turned out soon that the relational data model is not an ideal persistence solution for data warehouses. In operational relational databases data is, or at least should be, organized in third normal form which imposes constraints on the logical organization of data concerning primary key definition and redundancy. For more information on normal forms in relational databases see [29]. In OLTP (online-transaction-processing) systems it is crucial that data remains consistent over multiple updates, inserts and deletes of data records. Furthermore read as well as write operations on the database must be fast and scale well with respect to number of users and amount of data.

OLAP (online-analytical-processing) systems have different requirements and a different view on data. Different analysis need different “slices” of data from the data warehouse. A company manager for example could be interested in the total revenue of a specific office in a given timespan or in the sales of a given product in a specific region or in sales over the Internet to a specific age class. Every subject like product, region and customer can be seen as one dimension in the data model.

In a relational database in third normal form such queries would require complex sorting and expensive joining over various tables. Multidimensional data models are much better suited. In order to be suitable for OLAP a multidimensional database must support the following basic functionality [43].

Rotation Rotation allows switching between different views on a specific dimension. For example instead of querying for purchases of “products by age class” querying for purchases “by age class by product”.

Drill-Down and Roll-Up Drill-down means accessing more detailed data within a dimension. Roll-Up is the “inverse” operation. It allows accessing summarized data of a dimension.

Data Dice Dicing narrows the current view based on given characteristics. For example a query for products by age class can be filtered for females only. On this result set the operations rotation, drill-down, roll-up and dice must be available again.

4.5 Persistence strategies in data warehouses

The operations rotation, drill-down, roll-up and dice can either be simulated by a relational database (ROLAP) or a multidimensional database system (MOLAP) can be used. In any case the logical data organization is much different compared to operational relational databases.

ROLAPs use a relational database. They are suitable for large amounts of data with only a few number of dimensions [43]. Data is organized in a star- or snowflake schema [31]. The star- as well as the snowflake schema consists of one or more “fact” tables and several “dimension” tables. An example for a fact table is a table in which sales are stored. The fact table may contain information like the number of products sold and the earned profit. Information about the products themselves, or stores and regions where the products were sold is stored in the dimension tables which are linked to the fact table. Fact tables normally contains large numbers of records. To keep these records small detailed information is stored in the dimension tables. Dimension tables can have multiple hierarchies. A product for example may contain information about its brand. In the star schema such hierarchies are usually integrated into the dimension table. This implies that data has to be stored redundantly. In the snowflake schema denormalization is avoided by defining a table for each hierarchy. Using the snowflake schema is controversial because querying a normalized database requires a larger number of joins. Since joining is a very expensive operation redundancy and therefore increased storage requirements are often accepted.

Another mean to speed up queries is indexing. In OLTPs indexing is always a trade off between the speed of inserts and updates compared to the speed of reads. Therefore “over indexing” a table would severely slowdown the whole application because storing and retrieving have (at least for many applications) the same priority. In an OLAP on the contrary the time an update takes is not as important as the time to query the database and perform analysis. Therefore ROLAPs have in general a lot more indexes compared to an operational database.

Nevertheless ROLAPs only support a limited number of dimensions. If a high number of dimensions exists in the data warehouse MOLAPs are better suited [43]. They support a high number of dimensions and allow fast access. The trade off is that many parts of the data structure remain unused. Therefore the amount of data that can be stored is limited. To overcome these limitations HOLAPs can be used. Data that is accessed very often is stored in a multidimensional data structure whereas large amounts of data that are not accessed frequently are stored using ROLAPs.

4.6 Data extraction for data warehouses

The most common way to fill a data warehouse is a so called ETL process. ETL stands for “extraction-transformation-loading”. During the extraction phase data is loaded from different sources into a “staging area” where data is transformed in order to have a consistent representation of different sources. But not only data representation is changed. Data can be aggregated or enhanced by adding information from external sources. The transformed data is then loaded into the data warehouse. Since a data warehouse should be time variant and non-volatile ETL is normally done as a batch process. The more frequently the ETL process is performed the finer is the data granularity in the data warehouse. Finer data granularity also leads to more accurate analysis and therefore allows better decision support. On the other hand, the finer the data granularity the more data must be stored.

4.7 Types of data warehouses

Three concepts of data warehouses can be differentiated with respect to size and organisation [43].

4.7.1 Enterprise-wide data warehouse

First data warehouse solutions were built in the beginnings of 1990 with the intend to build an enterprise spanning data warehouse [23]. Every source available in a company should be extracted and stored in one central data warehouse. This concept soon turned out to be too inefficient and expensive. Including all data sources available in an enterprise resulted in very large amounts of data impossible to store and process in the beginnings of the 90’s.

4.7.2 Data marts

Like in many other fields of computer science the concept of divide and conquer seemed appropriate to manage these large amounts of data. The ideal solution of an enterprise spanning data warehouse was abandoned and solutions with a smaller scope were built. These solutions are called data marts and focus on a specific area of an enterprise like a department or on specific user or product groups. The architecture of a data mart is similar to that of a data warehouse.

Nevertheless building data marts as an alternative to an enterprise-wide data warehouse is controversial. In [34] David Marco analyses draw backs and shortcomings of independent data marts. A major problem he describes it that data marts are often built by different autonomous development teams. Therefore software and hardware varies between independent data marts. This implies that integrating independent data marts can be very cost and work intensive. Furthermore data is very often stored and processed redundantly, since data required for decision support for different business fields is likely to overlap. Customer data for example might be needed in several data marts. Since the ETL process is executed for each data mart the same data has to be cleaned and transformed several times. David Marco therefore suggests migrating from independent data marts to an enterprise-wide data warehouse consisting of dependent data marts [33].

4.7.3 Virtual data warehouse

In a virtual data warehouse data is not stored in a separate database. The data warehouse is only a layer that works as a middleware to execute queries on heterogeneous data sources. To the user it appears as if he queries one homogeneous database. For a long time virtual data warehouses were seen only as a temporary solution for prototyping or a solution for companies with a small budget [43]. The most significant drawback of a virtual data warehouse is performance. Since queries must be distributed over various source a user has to wait longer for the results. Furthermore a virtual data warehouse can slow down operational sources since data extraction can not be scheduled when the load on data sources is low. Therefore additional requests must be handled by operational data sources.

Nevertheless in the last years more and more attention is payed to virtual data warehouses and they have been reconsidered as an alternative.

4.8 Impacts of data warehousing

With the separation of operational and analytical systems the value of data increased. The way data is used in analytical systems differs fundamentally from usage in operational systems. Operational systems operate on data while analytical systems operate with data. Interestingly some aspects eliminated by relational databases reoccur in data warehouses. E.F. Codd introduced the relational data model to normalize data. OLAPs explicitly break some of these rules to fulfill the different requirements of analytical systems.

4.8.1 Data mining

With increasing processing power data warehousing became the basis for data mining. Data mining and data warehousing often goes hand in hand. The data warehouse provides data on which data mining tools can perform analysis. Due to the large amount of data it is very difficult for analysts to draw conclusions. Data mining algorithms are designed to find patterns in large amounts of data in order to predict trends. In [49] data mining is described as proactive process which tries to extract hidden predictive information from large databases.

Let's take a supermarket as an example where purchases are stored in an operational database. Information about these purchases can be extracted into a data warehouse. For example which products are bought together or, if customers can be identified by customer cards, information about which customer bought which products. Using data mining algorithms, user profiles or buying patterns and trends can be extracted from the data warehouse.

This is a further extension to a data warehouse since the data warehouse is now more than a system that provides collected business data. The difference is subtle but significant. Data warehousing enables the analyst to query the database in order to proof a hypotheses. Data mining algorithms are more powerful since they can automatically provide hypotheses for a given field of interest based on the data stored in the warehouse.

4.9 Evolution of data warehousing and data mining

Various evolutions in computer science contributed to the evolution of data warehousing and data mining. Most important were increased processing and data storage capacities but like in any other field of information science the evolution of the Internet had a great impact on data warehousing and data mining. Several aspects of data warehousing and data mining evolved. In order to examine impacts of the Internet it is important to look at the history as well as recent trends of the Internet.

5 Evolution of the Internet

5.1 The beginnings - ARPANET

The evolution of the Internet was a continuous process where the first milestone was the connection of four Universities in the United States in 1969. The so called ARPANET was a packet switching network and had a rather experimental character since the protocols for host to host communication were still under construction. The first available protocol was the Network Control Protocol introduced in 1970 [16].

Besides the ARPANET other private and public networks with different protocols and technical implementations evolved. The need for a uniform protocol increased. The DARPA (Defence Advanced Research Project Agency) funded a research project with the goal to find ways to connect these heterogeneous networks [6].

5.1.1 Birth of the Internet

It was in 1973 the TCP/IP protocol suite started to revolutionize network communication. In 1983 all ARPANET nodes used TCP/IP as a uniform communication protocol. The settlement on a standardized protocol was the basis for the evolution of the Internet. Therefore 1983 is often seen as the year of birth of the Internet [6].

5.1.2 First applications

The first applications aimed at resource sharing and provided means for long distance communication [16]. Telnet, FTP and e-mail were the first step towards collaboration over long distances although they did not change the “autistic” character of computer systems. Data exchange using FTP results in having a local copy of data. Files are not retrieved by a computer automatically but downloads are initiated by a user. The focus of these applications was human initiated (FTP) and human readable (e-mail) information exchange rather than inter application communication.

5.2 WEB

In 1991 Tim Berners-Lee and Robert Cailliau implemented the simple idea of documents stored on a server, accessible via a unique identifier. The document language proposed was HTML. In 1992 the Hyper Text Transfer Protocol was introduced. Having a uniform access protocol and file format which allows linking documents had many advantages over a central file server storing different file types. The most important advantage is that data can be accessed through a single application. It is therefore easier to access data and to check if data is still up to date because it is loaded from the server on demand. The possibility of linking documents means that more dynamic “connections” between data can be established. The primal intend of the Web was to help researchers at Cern in sharing information but the Web was very soon discovered as a platform for social and commercial use.

5.2.1 Architecture of the WEB

In the beginning the Web followed a strict Client-Server architecture. Documents hosted on servers were static HTML. The information flow was only from server to client. There was no active client participation. Active participation means that a client provides input that is either stored on the server and reflected to other clients or used to change the behaviour or appearance of a website. With the development of server side scripting languages it became possible to build dynamic web pages. Users were able to interact with the server. On the server side it was possible to use databases in order to generate dynamic HTML pages as well as to save user input. In 1998 about 30 Million hosts were connected to the Internet. Especially web shops like Amazon.com⁴ emerged [16].

5.2.2 Consequences for data mining

Information and file exchange via e-mail or FTP often results in having data distributed over various personal computers. Therefore it is a very demanding administrative task to keep such data consistent. Even if files are stored at a central file server consistency problem are likely to occur. The most common example is a lost update. If two people edit the same file both of them download the file from the server and work on a local copy. If both of them then upload their updated version back to the server without checking for concurrent modification one update is lost.

Data distributed over several personal computers and workstations is not accessible for machines and therefore not suitable for data warehousing and data mining. Even people working on a file concurrently by sending the file from one workstation to another have difficulties to keep track where the most recent version can be found.

Central file servers offer at least some possibility for automatic machine access. Nevertheless a problem that C.F. Codd addressed in [14] reoccurs - Access path dependence. An application trying to extract data from a file has to know the exact location and the exact name of the file. If the file is moved to another directory the application is not able to locate it anymore

With the evolution of the Web the concept of data administrated by a central authority but made available to a large number of clients located all over the world became common. Especially server side scripting and the usage of databases for persisting data contributed to data mining and warehousing since input data received from a large number of distributed clients is available in a central database.

5.3 Evolution from WEB to WEB 2.0

In [40] Tim O'Reilly describes the evolution of the Web after the burst of the .com bubble. Since several aspects of the Web 2.0 have an impact on data mining important the main ones are briefly summarized. In about 2001 a trend to move away from strict client server architecture

⁴<http://www.amazon.com>

started. Systems emerged where data and information is no longer administrated by a central authority only and many web applications relied on data supplied by users. This lead towards an architecture of participation.

5.3.1 Architecture of participation

Many successful web applications owe their success to the concept of user participation. They don't rely only on data managed by a central authority but emphasize users to contribute data and information. This way databases are filled with large amounts of data without the need for companies to provide data. Therefore large data collections are generated in a very cost effective manner. The online encyclopedia Wikipedia⁵ for example relies completely on data added by users. Amazon.com⁶ follows a mixed strategy. Like any other web book shop their service is based on "fact" data received from publishing companies. The key factor for success was that Amazon.com allowed users to participate by enhancing existing data. Users are allowed and encouraged to write recension that other users can read.

5.3.2 Social Networking

With the evolution towards WEB 2.0 social applications have become popular [51]. A well known example is Xing/openBC⁷. But many other platforms with the same concept but aiming at different user groups exist. The concept of these platforms is very simple. They allow their users to create a personal profile by entering personal and/or business data. Users can search for other users in order to add them as "contact" or "friend".

5.3.3 Data driven applications

Most of the successful WEB 2.0 applications are data driven. Tim O'Reilley concerning the value of software:

"The value of the software is proportional to the scale and dynamism of the data it helps to manage."

Tim O'Reilley [40]

Looking for example at social networking tools like Xing/openBC or web shops like Amazon.com it is not the software or the algorithms which are crucial for the success of the application. It is the data the application is based on that makes it attractive to users.

Centralized data management Therefore most Web 2.0 applications rely on centralized data management. Companies having more information and data available to present to their users are ahead of their competitors. Social portals like Xing/openBC are only attractive if many "real-life" friends or business contacts use the portal. The more content and information about content is available in a web shop the more users will be attracted.

⁵<http://www.wikipedia.org/>

⁶<http://www.amazon.com>

⁷<http://www.xing.com>

Peer to Peer systems emerging during the evolution to Web 2.0 were a counter movement to centralized data management. In fact the peer to peer architecture fits perfectly into the nature of the Internet which is not a centralized network but is based on “private” hosts which contribute to and enlarge the Internet. Peer to peer systems have the advantage that performance increases the more users participate. While database replication is difficult and expensive in common centralized client-server architectures in a peer to peer system data is replicated for free. One of the most successful peer to peer applications is BitTorrent. If such large amounts of downloadable content would be administrated by a central authority an administrator would be concerned with which data must be kept available and which content is no longer requested by users. In systems like BitTorrent data evolves naturally. Files and data that is no longer needed disappear “automatically”. But this implies that it is never known on which peers a specific file is currently available. Furthermore every peer can access all data available. Due to the increasing value of data, peer to peer systems are not widely used. One application is distributed and parallel processing. Data is still managed by a central authority, peers only receive the data necessary to perform a specific calculation.

5.3.4 Context independent data

One of the most important aspects of WEB 2.0 with respect to data management is the way data is used. In WEB 1.0 data was mostly collected and used in one specific context. Most information was available by pointing a browser to a specific URL in order to retrieve a static or dynamically created document. Therefore such data was accessible only within the context of a web page. WEB 2.0 applications have the characteristic that they are designed to make data accessible without the need to use a specific application. RSS allows users to check new posts and updates in a weblog with any application understanding the RSS protocol [44]. Information is separated from presentation and can therefore be easily integrated into third party applications. Tim O’Reilley describes this as:

“Design for hackability and remixability.”

Tim O’Reilly [40]

5.3.5 Service Orientated Architecture

SOA is a result and an “implementation” of context independent data. It enforces data independence on a very high level by allowing loosely coupling of so called services. Services can be applications as well as data sources. Loosely coupling is achieved by encapsulating a specific service behind a well defined interface. Aim of SOA is to allow interoperability between different software vendors, programming languages and paradigms. Neither for a client nor for a server is it important in which programming language their counterpart is written. Participating parties are only coupled by the interface description of the service. XML serves as a uniform file format for information transfer. The structure of the XML file is described in the interface of the service. An important advantage of SOA is that the service provider does not need to provide a customized front-end for every user of a service. Services should be designed to be context

independent. A client can integrate the functionality in any application without bothering the service provider. Google search⁸ for example is integrated in many third party websites. The University of Vienna⁹ for example uses Google to search within their Web page. The best known SOA standard are web services [11] and very often SOA has become synonymous with web services. However SOA is not a technology but a software architecture. Other frameworks like REST which is a lightweight web service framework based on the HTTP protocol are becoming more and more popular [18] [26].

Service orientated architecture is a trend away from classical client server architecture with one web server or server cluster offering the whole functionality for a web application. Many web applications are in fact portals composed of web services. These web services can be administrated by the same company that runs the web application but can as well be offered by external companies or providers.

5.4 Semantic Web

Most information in the Web especially the WEB 2.0 is human understandable only. All information on the Web that is accessible to a user is also accessible to machines. Although most data on the Web is machine processable it is not machine understandable [39]. With the evolution of SOA more and more data is available in the Internet that can be understood by machines. Since data is returned by web services as XML files described by an XML schema data can be parsed and processed automatically. Nevertheless XML is not the best way to generically represent information about data. Even if an application has access to an XML schema this does not necessarily imply that the semantics of the data are clear [10]. In many cases the semantics must be understood by an application developer and voven into the source code of an application. Furthermore applications which access web services have to use the interface definition provided by the web service. Since web service interfaces are not standardized they will look different, even for similar domains. Furthermore most web services are designed to provide functionality and not data access. Therefore most information on the Internet is still available in HTML pages. Although most web pages are dynamically created from structured machine readable sources like relational databases this structure is no longer available on the generated web page. Long before SOA emerged Tim Berners-Lee proposed a road map towards machine understandable information on the Web [9]. He describes the Semantic Web as

“a web of data, in some ways like a global database.”

Tim Berners-Lee [9]

In Semantic Web pages content is annotated with machine understandable and processable meta data. The Semantic Web is therefore a contribution to and not a replacement for the Web.

⁸<http://google.com>

⁹<http://www.univie.ac.at/>

5.4.1 Architecture of the Semantic Web

In order to understand the implications of the Semantic Web it is important to briefly look at its architecture. This section does not describe the Semantic Web framework in detail but gives a short overview about the concepts. For a more detailed introduction see [32].

RDF [39] is the basis of the Semantic Web architecture. It aims at adding meta data to resources in order to make them machine understandable. RDF itself is domain independent that means it is suitable to describe different resources using the same syntax. Basically RDF consists of Resources, Properties and Statements.

Resources are subjects like for example “web pages” or a “person” which are annotated with meta-data.

Properties are characteristics or attributes describing a resource like “author” for books or “lastname” for a person.

Statements annotate resources with properties and concrete values for the properties. For example: (“http://example.com”, host, exampleHost). The resources “http://example.com” is hosted on exampleHost. Statements can be nested. “exampleHost” for example could have properties like CPU- speed. Furthermore properties do not necessarily have a single value but can be instantiated with multiple values. A web page for example can be annotated with a list of mirrors. Property values can be literals, resources or statements [48].

RDF Schema [12] defines the “vocabulary” for RDF descriptions. It allows to create class hierarchies for resources and to define properties like in object orientated programming languages. RDF Schema is also used to define the usage of properties like that an “author” property can only be used to describe the relation between a “book” and a “person”. The RDF Schema language is domain independent. The purpose of an RDF Schema can be compared to the purpose an XML Schema. XML Schemata are themselves defined in XML syntax but only contain XML tags to describe the structure of an XML file. The same applies to an RDF Schema. It does not provide descriptions for application specific classes but RDF classes and properties to define these application specific classes.

Ontology languages are the next layer in the Semantic Web [48].

Ontologies are used to describe domain specific information in order to reuse knowledge. Domains are specific areas of knowledge like medicine. Ontologies contain computer processable information making data machine understandable and processable. In [27] goals for ontology languages are described. The, from my point of view, most important goals are summarized below.

Share ontologies It must be possible to publish and access an ontology description to allow applications to use these ontologies. Either to understand an existing data source which uses this ontology or to describe a data source using this ontology.

Evolution of ontologies Since an ontology might be modified and adapted over time, versioning is required to identify to which version of an ontology an application commits. Not only the formal structure of an ontology can change. The intended meaning of an identifier can change as well. For example a person's "id" property can in one version refer to a social insurance number and to a custom generated id in a later version. Therefore it must be possible to annotate semantic changes in order to guarantee backwards compatibility between versions.

Interoperability The same knowledge can be modeled by different ontologies. It must be possible to express relations between similar representations.

Inconsistency detection To prevent applications from integrating data sources which only pretend to commit to an ontology it must be possible to detect inconsistencies.

Internationalization Since the Web connects sources from all over the world one ontology must support different views on it with respect to language and culture.

The most common ontology language is OWL which is based on RDF and RDF Schema but provides additional vocabulary to describe properties and classes. For example constructs to mark different classes as similar. An introduction to OWL can be found in [35].

Logic The logic layer provides methods for reasoning. Very often this layer is integrated into the web ontology language because most ontology languages allow logical axioms. These logical axioms provide rules like "all authors must be persons" and "every person must have parents". Conclusions can be drawn based on these statements. For an author it can be concluded that he must have parents although no explicit statement like "every author has parents" exists. For logic reasoning there is always a trade off between expressivity and scalability [27]. The more complex logic axioms can be the more complex is reasoning. See [42] for reasoning issues on the Semantic Web.

6 Recent trends in data warehousing and data mining

6.1 EII - Enterprise Information Integration

As a consequence of the fast increasing number of additional data sources becoming available due to increased connectivity and loosely coupled applications, the idea of a virtual data warehouse has become a considerable alternative to large centralized data warehouses. Enterprise Information Integration aims at integrating different heterogeneous data sources without loading them in a centralized data warehouse. A virtual schema is built that can be queried by users in order to access the underlying data sources. An EII server has to decompose the queries and execute them on the available data sources. The results must be assembled and sent back to the user. One goal of EII is loosely coupling of data sources allowing on-demand integration of new data sources.

Several Web 2.0 aspects have contributed to the evolution of EII. Technologies providing context independent access like web services and RSS can be seen as precursors of EII. Portals created through loosely coupled data sources allow information integration on a global scale.

6.1.1 Differences between EII and EAI

EAI provides means for integrating, updating and orchestrating applications and data sources[24] (Michael Carey). So what's the difference to EII? In order to execute a query on distributed heterogeneous data sources EAI would require to create a business process to access the data sources. EII aims at

“[...] accessing locked data inside applications and web services [...]”

Michael Carey [24]

Instead of creating a business process by hand EII applications automatically create a query plan based on a declarative description of the data sources. For different query keys like search an employee “by ID”, “by department”, or “by project” different query plans might be needed if data is distributed over several databases. So EII systems can be used for fast and efficient queries in a virtual database. EII systems only support read operations since an update or insert operation requires a business process to manage aspects like long running transactions, validation checks and compensation activities.

Naveen Ashish [24] sees the core idea behind EII in moving away from a schema-centric data management approach. In a classical schema-centric approach data integrated from different sources is changed to fit in a predefined database schema of a data warehouse. If new sources need to be integrated the schema of the database must very often be adapted. Clients can query a server which executes the query and computes the results. By moving towards a schema-less approach it is possible to store data generically. No semantics or structure should be imposed on data by a server or a database. The client should be responsible for adding semantics to the data. This approach would allow clients with different requirements to query one and the same server. The results could be adapted to the needs of the querying client.

6.1.2 Advantages of EII

Real-Time Capabilities One of the biggest advantages of EII is that it gives users access to live data. The snap-shot concept resulting from the ETL process is eliminated. EII is therefore especially interesting for time critical applications where administrator must react on changing environments immediately like in for example stock trading applications.

Data does not have to be copied With increasing connectivity more and more large data sources become available. For applications which need to integrate many large data sources it is not feasible to extract all necessary information and copy it into a data warehouse. Although storage costs continuously decreasing EII systems are a considerable alternative if many large data sources must be integrated.

Easier integration of new data sources Due to the loosely coupling of data sources, several EII frameworks exist which allow integrating new data sources on-demand [24]. Some frameworks integrate data sources based on declarative descriptions. Another approach is integrating data sources by making them accessible through a web service. These specific web services are called data-services [7].

6.1.3 Disadvantages of EII

Performance One of the most striking disadvantages of EII is less performance. On the one hand queries are slower because they have to be decomposed and distributed to the data sources. The results of the single queries must be reassembled and sent back to the user. On the other hand EII can slow down the operational data sources themselves. For data warehousing the ETL process can be scheduled to a time when the load on operational data sources is low to reduce impacts on the performance of operation systems. Many queries performed by an EII system during a period of heavy load can result in unacceptable long response times for users of the operational system.

No historical data The real time capabilities of EII systems have the drawback that no historical data is available.

Accessibility problems Some third party systems may not be accessible for an EII system for security or performance reasons. So for some sources copying data into a company internal storage can not be avoided.

No complex analysis Since queries are federated over various sources data can not be pre-processed. In data warehouses data can be filtered and cleaned in the staging area. In EII systems data must be filtered and transformed for every query resulting in a higher demand on resources.

6.1.4 Advantages and Disadvantages - Implications for EII

Looking at the advantages and disadvantages of EII and data warehouses it seems that neither technology can replace the other. Dina Bitton [24] describes which approach is preferential for which requirements. If historical data is needed or complex resource intensive analysis must be performed storing data in a data warehouse is unavoidable. A virtual data warehouse on the other hand can be used to integrate data sources into existing data warehouses or data marts. Furthermore it can be used for prototyping or one time reports and of course if real-time capabilities are needed.

6.1.5 EII and Semantic Web

One of the big advantages of EII is that heterogeneous data sources can be integrated easily using data-services. Nevertheless integrating data-services faces a problem already known for web-service integration. Dynamic and automatic detection and integration of services without human intervention. A client application must be explicitly designed to understand and integrate a service.

“Information integration is doable - write enough code and I will connect every software system anywhere.”

Jeff Pollock [24]

Integrating Semantic Web principles into data services is an upcoming approach addressing this problem. Adding machine understandable meanings to data sources claims to be the clue for dynamically integrating services. In [7] a framework for this purpose is proposed and some important requirements for integrating the Semantic Web and EII are described. If agreed ontologies existed for specific domains, to which data sources from different enterprises and data suppliers committed data integration would be an easier task. Since this condition can not be taken for granted it is more likely that data integration must be done by mediated ontologies which translates between different ontologies from the same domain. The desired outcome is an

“[...] integration infrastructure that is dynamic, autonomous, self-organizing and proactive [...]”

Cheng Leong Ang and Robert Gay and Olaga Sourina [7]

In [7] key issues for large scale data integration are described. The, from my point of view, most important issues are described below.

Integration of ontologies from different domains Most applications need to integrate data from various domains. Since ontology creation is a difficult and time consuming process the mediated ontology should in the best case be created automatically. Nevertheless not every issue can be resolved by a software system automatically. Therefore a completely autonomous system can only be seen as a theoretical ideal case. In practice a human administrator is required to solve problems like incomplete or conflicting domain ontologies.

Handle ontology changes and evolution of existing data sources Software systems and data sources are continuously changing. To discover and integrate systems dynamically, changes in ontologies of data sources must be handled with few human user intervention. Integrating changes from domain ontologies into the mediated ontology raises the same problems as integrating new domain ontologies. In a large environment with many data sources it is not feasible to integrate every change by hand. Autonomous computer systems are needed to decide if human interaction is required.

Add and remove data sources dynamically Dynamic data-service discovery and integration would require a service registry with a large number of data-services with semantic descriptions. Autonomous agents would then be able to query the registry and find data sources for given field of interest.

Automatic generation and extraction of ontologies from data sources and applications

Handcrafting ontologies is a time consuming process. Due to the large number of data sources on the Web it is impossible to handcraft an ontology for each available data source. Therefore frameworks would be required to create ontologies from and for existing data sources. Research has been done for extracting information from structured data sources like relational databases over semi structured data sources like XML files and unstructured data sources like text files [7]. Problematically very often data sources are not directly accessible but hidden behind an application. Therefore it would be desirable to be able to extract ontologies from applications.

Automatic query processing and data extraction Queries formulated against a mediated ontology must be automatically distributed over various data sources. In the ideal case these data sources are discovered and queried in real-time. The ontology of a query is matched against ontologies of services in a semantically annotated service registry. Matching data-services are returned and the query is distributed to these services. Therefore the query must be rewritten to the language of the data source. The results must then be converted into a uniform representation in order to be returned to the querying application.

These requirements describe the ideal case for EII. Problematically ideal cases mostly don't work in practice. UDDI (Universal Description, Discovery and Integration) a universal web service registry created with the intend to support dynamic web service discovery and integration did not prevail in practice.

Even if it becomes possible to solve some of the very complex technical issues described above the problem of data authenticity remains. Without a trusted authority guaranteeing that a service commits to its semantic description the service or the service provider themselves must be "known" and trusted. But this severely reduces the possibilities and the flexibility of dynamic service discovery.

Nevertheless web services and service orientated architecture is now widely adopted. The same might happen to EII. The combination of EII and the Semantic Web seems promising, even if the ideal case might not work in practice. Due to the increasing number of large

and distributed data sources and the ever increasing demand for data integration research on technologies providing at least help in dynamic service discovery will continue.

6.2 Web mining - The World Wide Web as data source

The Web itself has become a valuable source of information. Very much information is contained in web pages or can be extracted by mining user behaviour. Especially for e-commerce applications web mining has become a key factor for being ahead of competitors. Web mining uses data mining methods and techniques and applies them to the World Wide Web [48]. In [45] four operations of interest are differentiated: clustering, classification, association and sequential analysis.

Clustering is used to group “similar” items. In a web shop for example clustering can be used to group products by analysing which products were bought together or to group users who have bought the same products or seem to have the same browsing behaviour.

Classification maps data items into predefined classes. Users for example can be mapped based on their personal data like age or gender or by preferences specified in their profile.

Association builds a relation between items and tries to predict correlation between them. For example users who searched for a product from group A might be interested in product group B as well.

Sequential analysis extends association rules by incorporating time. For analysing browsing behaviour it can be used to see which pages were accessed together within a given period of time. This allows very precise analysis. If a user for example browses over the pages A,B,C in a short period of time and some time after browses over the pages C,D,E it can be concluded that the pages A,B,C are related and the pages C,D,E are related. Without a time factor A,B,C,D,E might be grouped together because they were accessed by one user within the same session.

6.2.1 Subcategories of Web Mining

Web mining can be decomposed into three subcategories [48]:

- Web content mining
- Web structure mining
- Web usage mining

Content mining aims at extracting information from web sources like HTML pages [45]. Most available sources are semi-structured or unstructured which makes content mining more difficult than extracting information from a relational database where the schema is known. Content mining algorithms help in categorizing pages or products into groups without an administrator’s intervention.

Structure mining focuses on extracting information about how pages are linked. An example is the Google page rank algorithm which rates pages by how many other pages link to a page [48].

Web usage mining aims at finding patterns concerning user behaviour. It provides methods to evaluate implicit user data like click streams and integrate it with explicit user input like preferences and personal data. Web usage mining is a key factor for creating user profiles since users with similar behaviour can be grouped.

Especially powerful are combinations of the subcategories. Usage mining combined with content mining can provide associations between user groups and product categories. Structure mining combined with usage mining can be used to customize a web site for specific customers. If a user often requests pages in an order like A,C,D,B remaining on page A and B while C and D are only requested to browse to page B the user can be presented with a direct link from page A to page B. Besides being more convenient for the user and it also improves server performance since unnecessary requests for page C and D are omitted.

6.2.2 Web mining and Semantic Web

Since the Web is designed to be human readable web mining is a complicated task. Extending web data with machine understandable semantics contributes to and eases web mining tasks. To make the advantages of semantic web mining more comprehensible let's take a music web shop as an example. An ontology including notations like genres, sub-genres, interpreter or product-types (e.g.: DVD, CD,...) can be used to annotate pages and content with machine understandable semantics. Since products as well as hyperlinks on pages are annotated with semantics a link from a product page to an interpreter information page has a different annotation than for example a link to another product from the same interpreter. Mining can be done by evaluating application events a users creates navigating through a web site. In [48] it is differentiated between atomic and complex application events. Atomic application events are for example a single request for a specific product or the request of a specific page or service. Since most pages are created dynamically from a database analysing requests for a page is not enough. A search page for example might be accessed frequently. The interesting information lies in the service requested and the parameters provided by the users. There is a semantic difference whether a user searches for a Mozart opera or a Micheal Jackson DVD. Complex application events are sequential atomic events created by a user with the intend to solve a given problem or task. Mining these sequences allows finding patterns on a high level of abstraction. Users can not only be categorized by their interests but by their current intend. The browsing behaviour of users who want to inform themselves about products is different to the browsing behaviour of users who already know which product they want. If the user's intend can be discovered the page structure can be adapted to support the user reaching his goal. If a user's intend is knowledge building about products he can be presented with additional reviews, extended content description and other interesting information. On the other hand if a user aims at buying a specific product to much additional information might confuse the user and should therefore be omitted. Mining such events is not a new approach. So how does the Semantic Web contribute to and enhance web mining? Since content and pages are mapped to a concept hierarchy web mining is done

on these abstract concepts instead of concrete items. A product A is not only an entry in a database with some additional, implicit information that can be retrieved by following relations in the database. Content information like music genre, sub genre, interpreter or product category is explicitly attached to a product. Therefore conclusions on an arbitrary high level of abstraction can be drawn. Patterns like users “who searched for operas tend to buy musicals DVDs” can be found. Since mining is no longer done for concrete products new products can be recommended as well since they are described by their attached semantics which can be matched against existing user profiles. The same applies to page requests. Since dynamically created pages change their content and meaning for each request navigational patterns are difficult to mine if concrete pages are taken into account. But if service requests can be mapped to concepts, mining can be done on the abstract concept level.

User profiling The evolution of the Semantic Web might contribute especially to user profiling.

With the evolution of the Web 2.0 social platforms like XING/OpenBC have gained popularity. While people often tend to provide wrong or incomplete information about themselves when asked for their preferences and interests at e-commerce pages users enter detailed personal data at social community sites. Depending on the purpose of the community site, for example whether it is a business contact portal or a portal for personal contacts and friends users provide information about their job and their skills, their interests, hobbies and even their political attitudes. Additionally it can be found out who knows whom and the type of relation like “friend” or “business contact” between persons. With the available data a complete personal profile sometimes even the complete life of a person can be reconstructed. Mining these portals is therefore an valuable source of information. These user profiles are not only interesting for commercial purposes like personalized advertisements. For governments and military intelligence user profiles and especially connections between people are valuable information. By evaluating profiles governments hope to discover individuals who might be a threat to national security.

If community sites for example annotated their sites with machine understandable semantics and agreed on a uniform ontology these sites could be mined automatically. The FOAF¹⁰ (Friend-of-a-Friend) project for example introduces a vocabulary [13] based on RDF and OWL which allows to create machine readable descriptions of users, groups and companies. The base class of a FOAF description is a foaf:Agent. An agent can be for example a foaf:Group a foaf:Organisation or most common a foaf:Person. These classes can be associated with properties ranging from simple to very complex properties. Basic foaf:Person properties are for example firstname or lastname. Complex properties include personal information like attended schools or personal interests as well as information about foaf:OnlineAccounts. foaf:OnlineAccounts is a base class with concrete account types like foaf:OnlineEcommerceAccount for web shop accounts, foaf:OnlineChatAccounts or foaf:OnlineGamingAccounts. The most important property is foaf:knows which expresses that two persons know each other.

¹⁰<http://www.foaf-project.org>

Privacy and authenticity Since many social community sites are public accessible privacy issues are an important topic. Most social portals allow their users to select which data should be accessible for all users and which data should be accessible for authorized users only. Therefore users have at least some control over their data. FOAF aims at constructing a non proprietary social network where files are attached to web pages. Since these profiles are not managed by a central authority security, privacy and authenticity are important issues. Users must be able to restrict access to their profiles. Furthermore authenticity must be guaranteed. Otherwise it is easy to publish profiles with faked data in order to discredit someone. These issues are much more difficult to address in an open system like FOAF than in a proprietary social community portal.

6.3 Multimedia data mining

With the evolution of the Internet and increased processing and storage capacities many additional data sources became available. Besides semi-structured data like XML files many unstructured sources became available and interesting to mine. Especially in mining multimedia data much research has been done. In [50] Bhavani Thuraisingham et al. classify the following subcategories of multimedia data mining:

- Text Mining
- Audio Mining
- Image Mining
- Video mining

6.3.1 Text mining

Text mining is an important field in data mining because much information is available as human readable texts. A text is not necessarily saved as text document but can be available in a relational database like for example product descriptions for a web shop. Finding patterns in order to classify and associate texts is interesting for recommender systems. An example can be found in [17].

6.3.2 Audio mining

Audio mining is the field of finding patterns in audio streams. Audio mining is used in various fields, ranging from extraction of semantic information from audio streams in order to categorize its content, to finding abnormalities in audio surveillance data [38].

6.3.3 Image and video mining

Image and video mining require a separate treatment compared to other multimedia mining techniques [37]. Since images have a very complex representation large image databases require large amounts of storage capacities. For video mining additionally temporal relations between

images must be handled. Although video mining is a subcategory of multimedia data mining it contains aspects from image, text and audio mining.

Basically two techniques are used to represent images for image mining: iconic representations and symbolic representations [37].

Iconic representation The iconic representation is the original representation of the image, for example as two dimensional pixel array.

Symbolic representation Advanced mining techniques are based on image models. An image model uses a semantic description to represent an image. It is therefore possible to enrich image representations with additional semantics. Raw images and videos contain a lot of redundant information. Using an image model allows to eliminate these redundancies in order to reduce memory requirements. For videos the model must additionally contain information about temporal changes of the image. Using a symbolic representation means that mining is not done on raw image data anymore but on image entities annotated with semantics at an arbitrary level of abstraction. Simple standards like Exif or the IPTC-NAA-Standard annotating images with textual information already exist for several years. Such information includes keywords like the photographer's name or the location where the image was taken. These standards are incorporated into common image formats like JPEG. However symbolic image models can be annotated with very complex annotations like domain ontologies as well.

Image and video mining applications Image and video mining have become interesting for various applications and domains including medicine, monitoring systems, surveillance and commerce. A diagnostic decision support system for radiologist to identify aberration and diseases in x-ray images is a possible application for image mining. An example for commercial use of video data looks as follows:

Let's return to the supermarket example introduced in the section data warehousing. What an analyst misses mining operational data sources only, is information on which products a customer looked at before he decided to buy a specific product because such information is generally not available in a structured relational database. By tracking customers on video records it is possible to identify a customer's path through a supermarket discovering which products he looked at and how long he remained at a specific storage rack. If the customer can be identified this data can be associated to the products a customer finally bought. Therefore a shop owner can not only draw conclusions about which products are bought together but why a customer decided for a specific product.

6.4 Data streams

With increasing data transfer rates data sources different to common "static" data sources like relational databases have emerged. Much data interesting to mine arrives as stream. Nan Jiang and Le Gruenwald describe data streams as:

“[...] an ordered sequence of items that arrives in timely order. Different from data in traditional static databases, data streams are continuous, unbounded, usually come with high speed and have a data distribution that often changes with time.”

Nan Jiang and Le Gruenwald [28]

6.4.1 Streaming applications

Streaming applications can be found in most domains where computers systems are used. To see how widespread data streams have become some example data applications should be analysed.

Sensor networks Sensor data has become ubiquitous in various domains ranging from temperature sensors to various sensors integrated modern in cars. Sensor networks need to collect and evaluate data from different sensors which continuously send measurement results at a high data rate. Sensor networks can be very often found in monitoring applications. Some examples are:

Power plants Analysing power usage statistics to adjust power generation rates in power plants requires evaluating data collected from various streams [21] . Possibly weather sensors could be included to switch to windmills or solar power stations instead of for example coal power plants.

Electronic Stability Programs ESP in modern cars requires decisions based on various sensors including speed, current steering angle possible lateral drifting of the car.

GPS route guidance systems Many devices evaluate current GPS position data and traffic information from Internet sources to compute alternative routes in case of traffic jams.

Radio-frequency identification RFID chips can store information that can be retrieved contact-less. RFID chips are often used in supply chain management to identify and track products. Automatic tracking of products attached with RFID transponders from the factory over functional intermediaries to the shop where they are sold creates large data streams.

Multimedia data Video and audio data always arrives as stream. However evaluating such data can result in additional data streams. Let's refine the example of evaluating user behaviour in supermarkets by analysing video data. For tracking a customer several video streams have to be mined. Those have to be transformed into a stream of customer positions containing the required information for extracting patterns of behaviour for each individual customer.

Internet Sources Many information sources available in the Internet provide streamed data. Examples include financial tickers or news tickers. These sources provide data at a by far lower rate compared to sensor data or multimedia data streams. Most often data is available in a structured way making it easier to mine compared to unstructured data contained in for example video streams.

Click streams A user creates large amounts of data browsing over a web site. Since browsing over a web site is a continuous process data is available as click stream. Mining click stream data is especially interesting because it allows drawing conclusions about a user's interests and preferences. Joseph Rozenfeld [46] refers to such data as "haggling" data. In a web shop for example it such data can be used to draw conclusions about the reasons why a user decided for a specific product or why he cancelled an order instead of completing it. Click streams can be collected with various techniques. In server side and client side methods [47] are differentiated. One client side approach is null server logging can be used which works by embedding information in HTML tags which is sent to the server when the client interacts with the web page. The easiest server side method to analyse click streams is to evaluate the server log file. Analysing a server log does not have any impact on the server performance and can be realized without any or very few changes to the server. More sophisticated techniques like capturing TCP/IP packets exit which can capture data like how often did a user cancel a request and which pages are cancelled more often than others.

6.4.2 Issues in data stream mining

Due to fundamentally different characteristics of data streams compared to "static" data sources like relational databases additional requirements on memory management, communication and processing capabilities must be taken into account [19].

Memory management Many data streams interesting to mine create large amounts of data. RFID chips for example create such large amounts of data that persisting the whole stream in a database is not feasible [25]. An extreme example is the Large Hadron Collider (LHC), a particle accelerator and collider located at CERN¹¹. Experiments are supposed to start within 2007. In a press release CERN estimates the amounts of data produced per year with 12-14 petabytes [3]. But this is only the amount of which is stored and made available for processing. The data produced by the sensors is much higher. It arrives as a stream with a data rate in the range of 1 PB/s [8].

Communication capabilities Many streams are available from distributed or remote sources. Video streams from surveillance cameras might be analysed and mined at a data processing center. RFID data for a specific product is generated at different locations starting from the fabric where the product is manufactured to the shop where it is sold. Data that actually belongs to one specific stream but is created at different locations and must be sent to a central station in order to analyse it. If streams are very large high data transfer rates and a high bandwidth are required to transmit the stream.

Processing issues Common data mining algorithms require several passes over data. For many streams this is not applicable due to the high data rate of the incoming stream.

¹¹<http://www.cern.ch>

6.4.3 Stream classification

For data mining streams can basically be divided into two categories: offline and online streams [28].

Offline streams are characterized by regular bulk arrivals. Click streams for example can be extracted from server log files in a batch process. Mining offline streams is in general less demanding because a bulk of the stream can be processed by iterating over it several times. Nevertheless due to large amounts of data in offline streams, scanning the whole stream whenever a new bulk arrives is not feasible.

Online streams are streams where data arrives continuously. Sensor data, RFID streams or web sources like stock tickers are examples of online streams. Online streams are processed immediately and discarded afterwards. This means that the used algorithms must process the stream at least as fast as the data items arrive. Additional complexity comes from the need to combine and analyse data from different streams together in order to find specific patterns and trends. One example are sensor networks. For weather monitoring temperature, humidity and atmospheric pressure measurements from different sensors must be joined and analysed together. Another example of related data streams are news and stock tickers.

6.4.4 Real time stream processing

The biggest challenge in data stream mining is that in most cases it must be done in real-time. The need for real time mining of data streams is motivated by technical as well as “business” needs.

Technical needs come from the resource issues for data stream mining described above. Most data streams can not be persisted in their original form. In many sensor networks data comes in at such a high rate that persisting every measurement is not feasible. There are two possibilities to address this problem: realtime-mining and real-time warehousing

Real-time mining Real-time mining means that the incoming data is mined immediately. Mining algorithms are applied directly to the streams. Based on the extracted knowledge actions are taken or suggestions are provide to a user.

Real-time warehousing Real-time warehousing means filtering important data from the incoming stream and persisting it in a data warehouse for later analysis. The stream is processed in real-time but no immediate knowledge extraction is done. Mining is done on the extracted data in the warehouse and not directly on the stream.

Business needs Whether real-time mining or real-time warehousing can be used does not depend on technical issues only. Business needs are non technical reasons for deciding how to process streams. Many applications need to take actions based on incoming data streams in real-time.

Stock/News tickers If recommendations are not provided based on the most recent data available the recommendations are worthless. Therefore the fastest decision support provider will be ahead of his competitors. Of course there is always a trade off with the accuracy of recommendations. If predicted trends are inaccurate or incorrect they are useless no matter how fast they are available.

Sensor networks in cars Systems like ABS or ESP have to evaluate incoming data streams within milliseconds to take the appropriate actions fast enough.

Click streams Although click streams are usually mined offline from server log files, for some web applications it might be useful to find patterns in real time. For a web shop it might be interesting to change the structure of a page dynamically based on user behaviour. If for example a product is accessed very often and by many users a link to this product can be presented on the main page of a web site. Since the Web is a highly dynamic field where user interests can change very frequently adapting the page structure should be done in near real-time. If click streams are evaluated in batch mode every, let's say, three days the customization of the page might come too late due to changed customer interests.

RFID stream applications Mining RFID streams can for example contribute to more efficient supply chain management. An inventory management system might automatically order additional goods if the frequency at which a specific product is sold raises to prevent supply shortage. Incoming RFID streams must therefore be mined in near real-time to order new products fast enough.

For other applications timing constraints are less important. Examples are some specific RFID stream applications or scientific experiments.

RFID stream applications RFID streams have applications that don't necessarily require real-time mining. For long term supply chain management decision support RFID data must not be mined immediately at arrival time. To find for example bottle necks in the supply chain management process it is sufficient to analyse aggregated information extracted from the stream over a longer period of time.

Scientific experiments For many scientific experiments measurement evaluation is not time critical. Let's stick to the example of the Large Hadron Collider (LHC) at CERN. It is not necessary (and would not be possible) to mine incoming measurements from sensors in real-time. It is sufficient to extract relevant information from the streams to large databases. Due to the high amount of incoming data the extraction process must be done in real-time. The collected data can then be evaluated offline.

6.4.5 Stream processing algorithms

Resource and computational challenges in data stream processing are addressed with established statistical and computational approaches [20]. Two different algorithm categories for stream processing can be differentiated.

Data based solutions aim at either summarizing the whole data set or choosing a subset of the stream to be analysed. Techniques for summarizing the data stream include synoptic data structures and aggregation

Synoptic Data Structures Creating synoptic data structures means applying techniques for summarizing the whole data stream for further analysis. Since synopsis data structures do not contain all the information available in the original stream only approximate results can be obtained by mining algorithms.

Aggregation Statistical measures like means and variance are used to summarize the data stream. This approach suffers from the same problems as synoptic data structures. Mining algorithms do not operate on exact data and can therefore not produce exact results.

Techniques for selecting a subset of the original stream include sampling, load shedding and sketching.

Sampling Sampling means selecting items from the data stream to have a smaller sized representative of the data. Sampling in data streams faces challenges like finding distinct elements in one pass over a data stream or supporting “weighted” items in a stream.

Load Shedding Load shedding is the process of dropping sequences of items from a stream. This method can be used if the receiving system is not capable of handling all incoming items. Load shedding is difficult to use with data mining since it is very difficult to determine which items can be dropped and which items contain important information.

Sketching Sketching is the process of reducing the dimensionality of the incoming data stream. The main issue is to preserve the properties of the data items the mining algorithm operates on.

Task based solutions are described in [20] as

“[...]solutions that modify existing techniques or invent new ones in order to address challenges in data stream processing.”

Mohamed Medhat Gaber and Arkady Zaslavsky and Shonali Krishnaswamy [20]

Examples for task based solutions are sliding windows and algorithm output granularity.

Sliding Window This technique assumes that the user is more interested in the most recent data. Therefore detailed analysis is done only on the most recent items in the data stream. Older data stream items are included in the mining processed as summarized data.

Algorithm Output Granularity AOG is a dynamic approach to cope with fluctuating high data rates. The granularity of the mining result is determined by the data rate of the stream and the available memory and processing capacities of the mining device. If the mining device has enough resources results have a very fine granularity. If resources are limited or the data rate of the stream increases mining is still possible with an adapted result granularity.

6.4.6 Event Driven Architecture

A recent trend in SOA are event driven systems. A good introduction to Event Driven Architecture (EDA) can be found in the report “Event-Driven Architecture Overview” from Brenda Michelson [36]. To see how EDA is involved in data mining it is important to shortly summarize the most important aspects. Event driven architecture is a software architecture pattern allowing the transmission of events between loosely coupled software components and services. Basically EDA is a publish/subscribe system where event consumers can subscribe for events. If an event is produced the event is dispatched to the subscribed consumers. Events are “notable” things that happen inside or outside an application domain. An event should contain information necessary to be understood by the receiver. Such information can be the type, name or timestamp of the event. The receiver can analyse the event and trigger appropriate actions like invoking a service or notifying a user. EDA applications can generally be divided into three main categories:

- Simple Event Processing
- Stream Event Processing (SEP)
- Complex Event Processing (CEP)

Simple event processing applications are basically publish/subscribe systems focusing on subject or predicate-based filtering over individual events [52]. Examples are filtering for event type or specific values contained in attributes of the event.

Stream Event Processing deals with tasks like identifying “notable” events from event data streams or processing multiple related event data streams to identify meaningful patterns.

Complex Event processing is concerned with processing consecutive events for meaningful patterns. This includes “waiting” for events coming from different sources or having different event types. Such queries can run over a long period of time and can include the “non occurrence” of events.

For data mining in particular the last two categories are interesting. Especially when SEP and CEP are combined powerful decision support systems and knowledge discovery systems can be built based on EDA. Multiple event streams can be mined for complex event patterns by defining appropriate rules. A concrete example for which SEP and CEP are applicable is a system that mines RFID readings encoded as event streams. An example for RFID stream mining which has already been examined is supply chain management to predict and prevent supply shortfalls. Another example is a company realizing physical access control using RFID cards could define rules to detect movement abnormalities of employees (or at least their RFID access cards). Dynamic access rules can be developed based on the RFID events. If an employee enters a restricted area infrequently or at a time when he normally does not enter this area he might be required to enter a code to make sure that access is not granted to unauthorized persons. If on the other hand an employee enters an area very often in a short period of time the employee can be granted access without entering his code several times.

7 Summary

Knowledge and information extraction from existing data sources has been a field of interest through our entire history.

With the evolution of computer systems and electronic data management new opportunities for information extraction became available. Large amounts of data impossible to analyse for a person can nowadays be processed automatically by machines in order to extract knowledge and predict trends. Nevertheless it was a long way until computer systems became complex knowledge extraction and decision support systems.

The first computers were built to perform complex mathematical calculations. In the beginning of the 60's computer systems consisted of mainframes and punch cards were used as storage media. These computers had various limitations. First of all data was not immediately accessible. Therefore large amounts of valuable information theoretically existed but were not accessible.

First computer systems were used for daily operational business. But soon operational data sources were discovered as a valuable source of information for decision support. Due to different requirements for analytical systems compared to operational systems the concept of data warehouses was introduced by Bill Inmon. Data warehouses are created by extracting and copying data from operational sources to a separate database.

Increasing processing power allowed data mining algorithms to be executed on data warehouses. While querying data warehouses provides analysts with business data for decision support data mining goes one step further. Data mining algorithms provide analysts with hypotheses about possible trends based on data in the data warehouse.

Beside increased processing and storage capacities, it was the evolution of the Internet that contributed to data warehousing and data mining. Due to increased connectivity of computer systems and the possibility of machine to machine communication an increasing number of distributed, heterogeneous data sources became available. Enterprise Information Integration frameworks have emerged for integrating these sources into a virtual data warehouse. EII eases the integration of new data sources and reduces the amount of storage capacity needed to construct a data warehouse.

Service orientated architecture has influenced data warehousing as well. One approach for EII is integrating data sources made available as web services.

These so called data-services are also a result of the increasing importance of data. Using data-services the access to data sources can be restricted to specific users or customers. An upcoming trend, the Semantic Web might contribute to automatic, dynamic discovery of data sources.

The largest data source which has become available is the Web itself. Therefore web mining has become an important field.

Like for EII the Semantic Web also claims the potential to contribute to web mining. Web mining on semantically annotated web pages allows reasoning on a much higher level of abstraction. Information about web pages or content like product category or author of a web page is explicitly available for a specific domain. Therefore data mining can be done on these concepts rather than on concrete pages. Instead of patterns like “users interested in page A are also interested in page B” abstract patterns “like users interested in operas are interested in musicals as well” can be found.

Another recent trend is multimedia data mining. Data like text documents, audio streams, images and videos have to be analysed in an increasing number of domains. Especially video mining is a challenging task because it combines aspects from audio, image and text mining. To ease mining it has become common to use a symbolic representation of images which allows annotating data with semantics at an arbitrary level of abstraction.

The most important trend in data mining and data warehousing is the increasing demand for real-time or near real-time knowledge extraction which includes real-time warehousing as well as real-time data mining.

Real-time data warehousing means that data is extracted from sources into a data warehouse in real-time by eliminating the batch character of the ETL process. Queries are therefore always executed on the most recent data. Real-time data warehousing can be realized in several ways. One technology that can be used is EII which creates a virtual data warehouse that integrates several operational sources. Another way is listening to the same inputs as operational data sources and extracting information in real-time.

Real-time data mining means that data is mined without extracting and loading it into a data warehouse.

Real-time data mining and warehousing is especially important for stream mining. Due to the characteristics of data streams, processing and mining streams faces additional challenges compared to mining “static” data sources like operational databases. Due to the high data rate in streams memory, processing and communication capabilities must be addressed. For most streams it is not possible to store the whole stream in a data warehouse for later analyses. Depending on the domain most streams must therefore either be mined in real-time or interesting data items must be extracted in real-time.

One specific technology that can be used to mine data stream items encoded as events is Event Driven Architecture. Especially Event Stream Processing and Complex Event Processing can be combined to extract complex patterns from event streams.

Literatur

- [1] An introduction to the data warehouse. <http://www2.sims.berkeley.edu/courses/is206/f97/GroupD/datawarehouse.html>.
- [2] Sql history/overview. http://www.itworld.com/nl/db_mgr/05142001, May 2001.
- [3] Lhc computing grid goes online. <http://press.web.cern.ch/press/PressReleases/Releases2003/PR13.03ELCG-1.html>, September 2003.
- [4] Timeline of computer history. <http://www.computerhistory.org/timeline/?category=cmptr>, 2006.
- [5] Daniel Abramovitch and Gene Franklin. A brief history of disk drive control. *Control Systems Magazine, IEEE*, 22(3):28–42, June 2002.
- [6] Paul Alpar. *Kommerzielle Nutzung des Internet*. Springer, Berlin, 1998.
- [7] Cheng Leong Ang, Robert Gay, and Olga Sourina. Data integration for virtual enterprise in cyberworlds. In *CW '05: Proceedings of the 2005 International Conference on Cyberworlds*, pages 392–395, Washington, DC, USA, 2005. IEEE Computer Society.
- [8] Ruediger Berlich, Marcel Kunze, and Kilian Schwarz. Grid computing in europe: from research to deployment. In *ACSW Frontiers '05: Proceedings of the 2005 Australasian workshop on Grid computing and e-research*, pages 21–27, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.
- [9] Tim Berners-Lee. Semantic web road map. <http://www.w3.org/DesignIssues/Semantic.html>, September 1998.
- [10] Tim Berners-Lee. Why rdf model is different from the xml model. <http://www.w3.org/DesignIssues/RDF-XML.html>, September 1998.
- [11] David Booth, Hugo Haas, Eric Newcomer, Michael Champion, Chris Ferris, and David Orchard. Web services architecture. <http://www.w3.org/TR/ws-arch/>, February 2004.
- [12] Dan Brickley and R.V. Guha. Rdf vocabulary description language 1.0: Rdf schema w3c recommendation 10 february 2004. <http://www.w3.org/TR/2004/REC-rdf-schema-20040210/>, February 2004.
- [13] Dan Brickley and Libby Miller. Foaf vocabulary specification. <http://xmlns.com/foaf/0.1/>, January 2006.
- [14] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.
- [15] National Research Council. *Funding a Revolution: Government Support for Computing Research*, chapter 6, pages 157–186. National Academy Press, 1999.

- [16] National Research Council. *Funding a Revolution: Government Support for Computing Research*, chapter 7, pages 169–183. National Academy Press, 1999.
- [17] Carsten Felden and Peter Chameni. Recommender systems based on an active data warehouse with text documents. In *HICSS '07: Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, page 168a, Washington, DC, USA, 2007. IEEE Computer Society.
- [18] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, 2000.
- [19] M. Gaber, M. Krishnaswamy, and S. Zaslavsky. Adaptive mining techniques for data streams using algorithm output granularity, 2003.
- [20] Mohamed Medhat Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. Mining data streams: a review. *SIGMOD Rec.*, 34(2):18–26, 2005.
- [21] Lukasz Golab and M. Tamer Oezsu. Issues in data stream management. *SIGMOD Rec.*, 32(2):5–14, 2003.
- [22] Jim Gray. Evolution of data management. *Computer*, 29(10):38–46, 1996.
- [23] Michael Haisten. The real-time data warehouse. the next stage in data warehouse evolution. <http://www.damanconsulting.com/company/articles/dwrealtime.htm>, August/September 1999.
- [24] Alon Y. Halevy, Naveen Ashish, Dina Bitton, Michael Carey, Denise Draper, Jeff Pollock, Arnon Rosenthal, and Vishal Sikka. Enterprise information integration: successes, challenges and controversies. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 778–787, New York, NY, USA, 2005. ACM Press.
- [25] Mark Hall. Databases can't handle rfid. <http://www.computerworld.com/databasetopics/data/story/0,10801,99589,00.html>, February 2005.
- [26] Hao He. Implementing rest web services: Best practices and guidelines. <http://www.xml.com/pub/a/2004/08/11/rest.html>, August 2004.
- [27] Jeff Heflin. Owl web ontology language use cases and requirements. <http://www.w3.org/TR/2004/REC-webont-req-20040210/>, February 2004.
- [28] Nan Jiang and Le Gruenwald. Research issues in data stream association rule mining. *SIGMOD Rec.*, 35(1):14–19, 2006.
- [29] William Kent. A simple guide to five normal forms in relational database theory. *Commun. ACM*, 26(2):120–125, 1983.

- [30] Ralph Kimball. Meta meta data data. making a list of data about metadata and exploring information cataloging tools. <http://www.fortunecity.com/skyscraper/oracle/699/orahtml/dbmsmag/9803d05.html>, March 1998.
- [31] Michael Krippendorf and Il-Yeol Song. The translation of star schema into entity-relationship diagrams. In *DEXA Workshop*, pages 390–395, 1997.
- [32] Frank Manola and Eric Miller. Rdf primer w3c recommendation 10 february 2004. <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>, February 2004.
- [33] David Marco. Data mart migration. http://www.dmreview.com/article_sub.cfm?articleId=256, December 1998.
- [34] David Marco. Stranded on islands of data. http://www.dmreview.com/article_sub.cfm?articleId=299, November 1998.
- [35] Deborah L. McGuinness and Frank van Harmelen. Owl web ontology language overview. <http://www.w3.org/TR/2004/REC-owl-features-20040210/>, February 2004.
- [36] Brenda M. Michelson. Event-driven architecture overview. <http://soa.omg.org/Uploaded%20Docs/EDA/bda2-2-06cc.pdf>, February 2006.
- [37] Rokia Missaoui and Roman M. Palenichka. Effective image and video mining: an overview of model-based approaches. In *MDM '05: Proceedings of the 6th international workshop on Multimedia data mining*, pages 43–52, New York, NY, USA, 2005. ACM Press.
- [38] Simon Moncrieff, Svetha Venkatesh, and Geoff West. Online audio background determination for complex audio environments. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(2):1–30, 2007.
- [39] Ralph R. Swick Ora Lassila. Resource description framework (rdf) model and syntax specification. <http://www.w3.org/TR/PR-rdf-syntax>, January 1999.
- [40] Tim O'Reilly. What is web 2.0. <http://www.oreillynet.com/lpt/a/6228>, September 2005.
- [41] Ken Orr. Data warehousing technology. <http://www.kenorrinst.com/dwpaper.html>, 2000.
- [42] Peter F. Patel-Schneider and Ian Horrocks. Position paper: a comparison of two modeling paradigms in the semantic web. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 3–12, New York, NY, USA, 2006. ACM Press.
- [43] Helge Petersohn. *Data Mining. Verfahren, Prozesse, Anwendungsarchitektur*, chapter 1 and 2, pages 4–57. Oldenbourg Wissenschaftsverlag GMBH, 2005.
- [44] Mark Pilgrim. What is rss. <http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html>, December 2002.

- [45] Nivedita Roy and Tapas Mahapaatra. Web mining: A key enabler in e-business. In *Proceedings of ICSSSM '05. International Conference on Services Systems and Services Management*. IEEE, 2005.
- [46] Joseph Rozenfeld. Unstructured data: Reading between the lines. http://www.dmreview.com/article_sub.cfm?articleID=1075145, February 2007.
- [47] Arun Sen, Peter A. Dacin, and Christos Pattichis. Current trends in web data analysis. *Commun. ACM*, 49(11):85–91, 2006.
- [48] Gerd Stumme, Anreas Hotho, and Bettina Berendt. Usage mining for and on the semantic web, 2004.
- [49] Kurt Thearling. An introduction to data mining. discovering hidden value in your data warehouse. <http://www.thearling.com/text/dmwhite/dmwhite.htm>.
- [50] Bhavani Thuraisingham, Chris Clifton, John Maurer, and Marion G. Ceruti. Real-time data mining of multimedia objects. In *Fourth International Symposium on Object-Oriented Real-Time Distributed Computing*, 2001.
- [51] Win Treese. Web 2.0: is it really different? *netWorker*, 10(2):15–17, 2006.
- [52] Eugene Wu, Yanlei Diao, and Shariq Rizvi. High-performance complex event processing over streams. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 407–418, New York, NY, USA, 2006. ACM Press.