Data-Warehouse-, Data-Mining- und OLAP-Technologien

Chapter 2: Data Warehouse Architecture

Bernhard Mitschang
Universität Stuttgart

Winter Term 2014/2015
Overview

• Data Warehouse Architecture
  ▪ Data Sources and Data Quality
  ▪ Data Mart
  ▪ Operational Data Store

• Information Integration
  ▪ materialized vs. virtual
  ▪ Federated Information Systems

• Metadata
  ▪ Metadata Repository
  ▪ Metadata in Data Warehousing
  ▪ CWM Metamodel
Architecture

- **Data Warehouse Architecture**
- **Data Staging Area**
  - Extraction
  - Transformation
  - Monitor
- **Data Warehouse Manager**
- **Metadata Manager**
- **Metadata Repository**
- **Data Warehouse**
- **End User Data Access**
- **Data Warehouse System**
  - Load
  - Data Sources
  - Data Staging Area
  - Control flow
  - Data flow

(Source: [BG04])
Data Sources

• Characteristics of source systems:
  - narrow, "account-based" queries
  - no queries in a broad and unexpected way, like DW
  - maintain little historical data
  - no conformed dimensions (product, customer, geography, ...) with other legacy systems
  - use keys (production keys) to make certain things unique (product, customer, ...)

• Important issues in selecting data sources:
  - Purpose of the data warehouse
  - Quality of data sources
  - Availability of data sources (organizational prerequisites, technical prerequisites)
  - Costs (internal data, external data)
## Data Quality

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>consistency</strong></td>
<td>Are there contradictions in data and/or metadata?</td>
</tr>
<tr>
<td><strong>correctness</strong></td>
<td>Do data and metadata provide an exact picture of the reality?</td>
</tr>
<tr>
<td><strong>completeness</strong></td>
<td>Are there missing attributes or values?</td>
</tr>
<tr>
<td><strong>exactness</strong></td>
<td>Are exact numeric values available?</td>
</tr>
<tr>
<td></td>
<td>Are different objects identifiable? Homonyms?</td>
</tr>
<tr>
<td><strong>reliability</strong></td>
<td>Is there a Standard Operating Procedure (SOP) that describes the provision of source data?</td>
</tr>
<tr>
<td><strong>understandability</strong></td>
<td>Does a description for the data and coded values exist?</td>
</tr>
<tr>
<td><strong>relevance</strong></td>
<td>Does the data contribute to the purpose of the data warehouse?</td>
</tr>
</tbody>
</table>
Dimensions of Data Sources

- **origin**
  - internal vs. external data

- **time**
  - current vs. historic data

- **usage**
  - data vs. metadata

- **type**
  - number, string, time, graphic, audio, video, ... numeric, alphanumerical, boolean, binary, ...

- **character set**
  - ASCII, EBCDIC, UNICODE, ...

- **orientation**
  - left to right, right to left, top-down

- **confidentiality**
  - strictly confidential, confidential, public, ...
**Monitoring**

- **Goal:** Discover changes in data source incrementally
- **Approaches:**

<table>
<thead>
<tr>
<th></th>
<th>Based on ...</th>
<th>Changes identified by ...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trigger</strong></td>
<td>triggers defined in source DBMS</td>
<td>trigger writes a copy of changed data to files</td>
</tr>
<tr>
<td><strong>Replica</strong></td>
<td>replication support of source DBMS</td>
<td>replication provides changed rows in a separate table</td>
</tr>
<tr>
<td><strong>Timestamp</strong></td>
<td>timestamp assigned to each row</td>
<td>use timestamp to identify changes (supported by temporal DBMS)</td>
</tr>
<tr>
<td><strong>Log</strong></td>
<td>log of source DBMS</td>
<td>read log</td>
</tr>
<tr>
<td><strong>Snapshot</strong></td>
<td>periodic snapshot of data source</td>
<td>compare snapshots</td>
</tr>
</tbody>
</table>
Data Staging Area:
A storage area and a set of processes that clean, transform, combine, deduplicate, household, archive, and prepare source data for use in the data warehouse.


- Data is temporarily stored in the data staging area before it is loaded into the data warehouse.
- All transformations are performed in the DSA.
  - Preprocessing does not influence data sources or data warehouse
- DSA is the central repository for ETL (Extraction - Transformation - Load) processing.
Extraction

- Transfer data from data source into the data staging area.
- Extracted subset of data sources and schedule of the extraction depends on the kind of analysis that should be supported.
- Method depends on the monitoring strategy used:
  - Read data from a file written by triggers.
  - Read data from replication tables.
  - Select data based on the timestamp.
  - Read data from log.
  - Read output of snapshot comparison.
- Multiple extract types:
  - periodic
  - started by the admin/user
  - event-driven
  - immediate after changes in data sources
Transformation

- Convert the data into something representable to the users and valuable to the business.
  - Transformation of structure **and** content

- Typical transformations:
  - denormalization, normalization
  - data type conversion
  - calculation, aggregation
  - standardization of strings and date values
  - conversion of measures
  - cleansing (missing, wrong, and inconsistent values)
Load

- Transfer data from the data staging area into the data warehouse.
- Data in the warehouse is rarely replaced. The history of values/changes is stored instead.
- Mainly based on bulk load tools of the DBMS.
- Offline vs. online load.
- Parallel load may be required.
Data Warehouse Manager

- Controls all components of the data warehouse system:
  - **Monitor**: Discover changes in data sources
  - **Extraction**: Select and transfer data from data sources to the data staging area
  - **Transformation**: Consolidate data
  - **Load**: Transfer data from data staging area to the data warehouse
  - **End User Data Access**: Analysis of data in the data warehouse
Basic Elements of the Data Warehouse

Source Systems

Storage:
- Flat files;
- RDBMS;
- other

Processing:
- clean;
- prune;
- combine;
- remove duplicates;
- household;
- standardize;
- conform dimensions;
- store awaiting replication;
- archive;
- export to data marts;

No user query services

Data Staging Area

Storage:
- Flat files;
- RDBMS;
- other

Processing:
- clean;
- prune;
- combine;
- remove duplicates;
- household;
- standardize;
- conform dimensions;
- store awaiting replication;
- archive;
- export to data marts;

No user query services

"The Data Warehouse"

Presentation Servers

Data Mart #1:
- OLAP query services;
- dimensional!
- subject oriented;
- locally implemented;
- user group driven;
- may store atomic data;
- may be frequently refreshed;
- conforms to DW Bus

Data Mart #2:
- Conformed dimensions
- Conformed facts

Data Mart #3:
- Conformed dimensions
- Conformed facts

End User

Data Access

Ad Hoc Query
Tools

Report
Writers

End User
Applications

Models
- forecasting;
- scoring;
- allocating;
- data mining;
- other downstream systems;
- other parameters;
- special UI

(Source: [KR+98])
**Data Warehouse Architecture**

**Architecture**

**Central architecture**
- only one data model
- performance bottleneck
- complex to build
- easy to maintain

**Federal architecture**
- logically consolidated
- separate physical databases that store detailed data
- faster response time

**Tiered architecture**
- physical central data warehouse
- separate physical databases that store summarized data
- faster response time

(Source: [J L+02])
Data Marts

dependent data marts

independent data marts
Data Marts

Dependent data marts (tiered architecture)

- Central data warehouse (DW) is build first
- Extracts of the data warehouse are provided as data marts (materialized views)
- Establish ETL process for DW only
- Consistent analysis on DW and DM

Independent data marts (federated architecture)

- Several data marts (DM) are build first
- Data marts are integrated by means of a second transformation step
- Establish ETL process for each DM and the central DW
- Inconsistent analysis is possible
- Virtual data warehouse possible (federated architecture)
Operational Data Store (ODS)

- Term has taken many definitions. For example:
  - **Point of integration for operational systems**
    - refreshed within a few seconds after the operational data sources are updated
    - very little transformations are performed
    - Example: Banking environment where data sources keep individual accounts of a large multinational customer, and the ODS stores the total balance for this customer.
  - true operational system separated from the data warehouse

  - **Decision support access to operational data**
    - integrated and transformed data are first accumulated and then periodically forwarded to the ODS
    - involves more integration and transformation processing
    - Example: Bank that stores in the ODS an integrated individual bank account on a weekly basis
  - part of the data warehouse or separate system?
Classes of Operational Data Stores

- Tables are copied from the operational environment.
- Transactions are moved to the ODS in an immediate manner (range of one to two seconds).
- Activities in the operational environment are stored, integrated, and forwarded to the ODS.
- ODS is fed aggregated analytical data from the data warehouse.
- Combination of integrated data from the operational environment and aggregated data from the analytical environment.

Overview

• Data Warehouse Architecture
  ▪ Data Sources and Data Quality
  ▪ Data Mart
  ▪ Operational Data Store

• Information Integration
  ▪ materialized vs. virtual
  ▪ Federated Information Systems

• Metadata
  ▪ Metadata Repository
  ▪ Metadata in Data Warehousing
  ▪ CWM Metamodel
• **Definition**: Federated Information System

A federated information system consists of a set of distinct and autonomous system components, the participants of this federation. Participants in first place operate independently, but have given up some autonomy in order to participate in the federation.

(Source: [BK+99])
Autonomy of Information Systems

- Design autonomy
  - The design of one source system is independent from the design of other systems, e.g., in the universe of discourse, the data model, naming concepts, ...

- Communication autonomy
  - Source systems decide independently with which other systems they communicate.
  - Source systems can leave and enter a federation at any time.

- Execution autonomy
  - Source systems independently decide on execution and scheduling of incoming requests.
Heterogeneity in Information Systems

- **Syntactical**
  - Technical heterogeneity
  - Language heterogeneity
  - Query restrictions
  - Binding restrictions
  - Data model heterogeneity
  - Semantic heterogeneity
  - Schematic heterogeneity
  - Structural heterogeneity

(Source: [BK+99])
Federated Information Systems

Federation layer
- e.g. uniform access language, uniform access schema, uniform metadata set

Foundation layer (data sources)
- Local Applications

Presentation layer

Wrapper layer
## Types of Federated Information Systems

<table>
<thead>
<tr>
<th>Types of heterogeneity addressed</th>
<th>Loosely coupled Information Systems</th>
<th>Federated Databases</th>
<th>Mediator-based Information Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only technical and language heterogeneity</td>
<td>All, except query restriction heterogeneity; schema integration difficult for schematic heterogeneity</td>
<td>All, except query restriction heterogeneity; schema integration difficult for schematic heterogeneity</td>
<td>All</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Loss of autonomy?</th>
<th>Execution autonomy</th>
<th>Execution autonomy; notification of schema changes</th>
<th>Execution autonomy</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Transparency</th>
<th>Language</th>
<th>Location, schema and partly language</th>
<th>Location, schema and language</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Kind of components</th>
<th>Structured</th>
<th>Structured</th>
<th>Any</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Access methods</th>
<th>Query language</th>
<th>Query language</th>
<th>Any</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Access restrictions</th>
<th>No</th>
<th>No</th>
<th>Yes</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Write access?</th>
<th>Yes</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
</table>
# Types of Federated Information Systems

<table>
<thead>
<tr>
<th></th>
<th>Loosely coupled Information Systems</th>
<th>Federated Databases</th>
<th>Mediator-based Information Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tight vs. loose integration</strong></td>
<td>Loose</td>
<td>Tight</td>
<td>Tight</td>
</tr>
<tr>
<td><strong>Kinds of semantic integration</strong></td>
<td>Collection</td>
<td>Collection and fusions</td>
<td>Collection, fusion, sometimes abstraction</td>
</tr>
<tr>
<td><strong>Necessary metadata</strong></td>
<td>Technical, infrastructure</td>
<td>Logic, technical, semantic</td>
<td>Logic, technical, semantic</td>
</tr>
<tr>
<td><strong>Bottom-up vs. top-down</strong></td>
<td>n.a.</td>
<td>Bottom-up</td>
<td>Top-down</td>
</tr>
<tr>
<td><strong>Virtual vs. materialized</strong></td>
<td>Virtual</td>
<td>Virtual</td>
<td>Virtual</td>
</tr>
<tr>
<td><strong>Evolvability</strong></td>
<td>High</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

(Source: [BK+99])
Mediator-based Information Systems

(Source: [BK+99])
Federated Database Systems

- Federated DBMS (FDBS)
  - Transparent access to a collection of heterogeneous and semi-autonomous data sources.

- Complete, extensible database engine
  - Function compensation
  - Powerful (global) query optimizer (pushdown analysis, cost-based optimization, query rewrite)

Federation layer

- e.g. uniform access language, uniform access schema, uniform metadata set

Presentation layer

Foundation layer

(data sources)
Architecture for a FDBS

Client

SQL API

Federated Database Server

Wrapper

Back-end Data Source

Data

Catalog

Data

Back-end Data Source

Data
Information Integration

Data Warehouse Architecture

Data Sources → ETL → Integrated Schema

End User Data Access

Query Execution

Global Schema → Schema Mappings → Wrapper

Mediator

Data Sources

materialized II

virtual II

(Source: [LN07])
Overview

• Data Warehouse Architecture
  ▪ Data Sources and Data Quality
  ▪ Data Mart
  ▪ Operational Data Store

• Information Integration
  ▪ materialized vs. virtual
  ▪ Federated Information Systems

• Metadata
  ▪ Metadata Repository
  ▪ Metadata in Data Warehousing
  ▪ CWM Metamodel
A repository is a shared database of information about engineering artifacts, such as software, documents, maps, information systems, and manufactured components and systems.

Functions of a repository:
- Object management
- Dynamic extensibility
- Relationship management
- Notification
- Version management
- Configuration management

(Source: [Ber98])
Metadata in Data Warehousing

- **Data dictionary:** Definitions of the databases and relationship between data elements
- **Data flow:** Direction and frequency of data feed
- **Data transformation:** Transformations required when data is moved
- **Version control:** Changes to metadata are stored
- **Data usage statistics:** A profile of data in the warehouse
- **Alias information:** Alias names for a field
- **Security:** Who is allowed to access the data

Stored in a metadata repository

Need for a standard interchange format
**Metadata in Data Warehousing**

- Criteria to identify important classes of metadata in data warehousing:
  - Type of data
  - Abstraction
  - User
  - Origins
  - Time

- Usage of metadata in data warehousing:
  - passive
  - active
  - semi-active

- Main goals:
  - Support development and operation of a data warehouse
    - system integration
    - processes for DW administration
    - flexible application development
    - access rights
  - Provide information for data warehouse users
    - quality of data
    - consistent terminology
    - support for data analysis
Centralized Metadata Management

![Diagram of Centralized Metadata Management]

- Monitoring Tool
- ETL Tool
- Data Warehouse Manager
- Data Access Tool

- Metadata Manager
- Metadata Repository

Arrows indicate:
- Data flow
- Control flow
Decentralized Metadata Management

meta data integration via point-to-point bridges

- Monitoring Tool
  - MD Manager
  - Metadata Repository

- ETL Tool
  - MD Manager
  - Metadata Repository

- Data Warehouse Manager
  - MD Manager
  - Metadata Repository

- Data Access Tool
  - MD Manager
  - Metadata Repository

(data flow)

(control flow)

(Source: [PC+03])
Federated Metadata Management

Diagram showing the components of a federated metadata management system, including:
- Monitoring Tool
- ETL Tool
- Data Warehouse Manager
- Data Access Tool
- Local Tool
- Metadata Repository

The diagram illustrates the flow of data and control between these components, with direct and indirect connections.

(Source: [BG04])
Metadata Interchange

- Standardization Approaches:
  - Case Data Interchange Format (CDIF)
    - Electronic Industries Association (EIA)
    - Focus on CASE Tools
    - Interchange based on files
  - Open Information Model (OIM)
    - Meta Data Coalition's Metadata Model
    - Interchange based on XML
    - Merged with CWM since 1999
  - Common Warehouse Model (CWM)
    - Metadata Model published by the Object Management Group (OMG)
    - Interchange based on XML
CWM Metamodel

The main purpose of CWM is to enable easy interchange of warehouse and business intelligence metadata between warehouse tools, warehouse platforms and warehouse metadata repositories in distributed heterogeneous environments.

- Design goals: The CWM model should …
  - reuse concepts in the UML metamodel where applicable.
  - be subdivided into packages allowing implementation of relevant subsets of the model.
  - be independent of any specific data warehouse implementation. Yet it should contain features that are effective in, and mappable to, a broad range of representative warehouse configurations based on specific tools.
Model-based Metadata Management

- Monitoring Tool
- ETL Tool
- Data Warehouse Manager
- Data Access Tool

- MD Manager
- Metadata Repository

CWM Metamodel

Data flow

Control flow

(Source: [PC+03])
CWM Metamodel

- CWM is based on three key industry standards:
  - UML - MOF - XMI
- CWM needs a formal language capable of representing meta data in terms of shared platform-independent models.
- CWM is based on a framework that supports any kind of meta data and allows new kinds of meta data to be added as required.
- Meta data integration based on CWM requires a common interchange format for exchanging instances of shared meta data.
The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components. (OMG)

- **Class Model**

<table>
<thead>
<tr>
<th>Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>x: Real</td>
</tr>
<tr>
<td>y: Real</td>
</tr>
<tr>
<td>rotate(angle: Real)</td>
</tr>
</tbody>
</table>

- **Inheritance**

  - Shape
    - Ellipse
    - Polygon
    - Spline

- **Composition**

  - Window
    - Slider
    - Header
    - Panel
    - scrollbar: 2
    - title: 1
    - body: 1

- **The UML notation is used in the diagrammatic representations of the CWM metamodel.**

(Source: www.omg.org/technology/uml/)
Meta Object Facility (MOF)

- The MOF model can be used as a model for defining information models.
- In this context, the MOF Model is referred to as a meta-metamodel because it is being used to define metamodels (UML, CWM, ...).
- The main modeling concepts are:
  - **Classes**, which model MOF meta-objects.
  - **Associations**, which model binary relationships between meta-objects.
  - **DataTypes**, which model other data (e.g. primitive types, external types, etc.).
  - **Packages**, which modularize the models.

Metadata Architecture

1. UML Model
   - model, metadata
2. UML Metamodel
   - metamodel, meta-metadata
3. The "MOF Model"
   - metametamodel
4. Modelled system
   - object, data

XML Metadata Interchange (XMI)

- Supports interchange of MOF-based meta-data.
- Applicable to a wide variety of objects: UML, Java, C++, EJB, IDL, CWM.
- Production of XML Schemas starting from an object model.
- Production of XML documents starting from objects.

# CWM Metamodel

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Mining</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Visualization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business Nomenclature</td>
</tr>
<tr>
<td>Resource</td>
<td>Object Model</td>
<td>Record</td>
</tr>
<tr>
<td></td>
<td>Relational</td>
<td>Multidimensional</td>
</tr>
<tr>
<td></td>
<td>Record</td>
<td>XML</td>
</tr>
<tr>
<td>Foundation</td>
<td>Business Information</td>
<td>Data Types</td>
</tr>
<tr>
<td></td>
<td>Expression</td>
<td>Keys and Indexes</td>
</tr>
<tr>
<td></td>
<td>Keys and Indexes</td>
<td>Type Mapping</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Software Deployment</td>
</tr>
<tr>
<td></td>
<td>Object Model</td>
<td></td>
</tr>
</tbody>
</table>

(Source: [PC+03])
CWM Relational Package

Data Warehouse Architecture

Anwendersoftware
CWM Warehouse Process Package
Summary

• Basic Components:
  - Data Staging Area: Extraction, Transformation, Load
  - Data Warehouse Database
  - Data Warehouse Manager
  - Metadata Repositories and Metadata Manager

• Data Marts: Distributed Data Warehouse

• Materialized vs. virtual information integration

• Metadata is important to:
  - Support development and operation of a data warehouse
  - Provide information for data warehouse users

• Metadata standards are important to interchange metadata between warehouse tools, warehouse platforms and warehouse metadata repositories.
Papers & Books


