Agility meets Data Warehouse Design

Stefano Rizzi
DISI – University of Bologna
stefano.rizzi@unibo.it

Summary

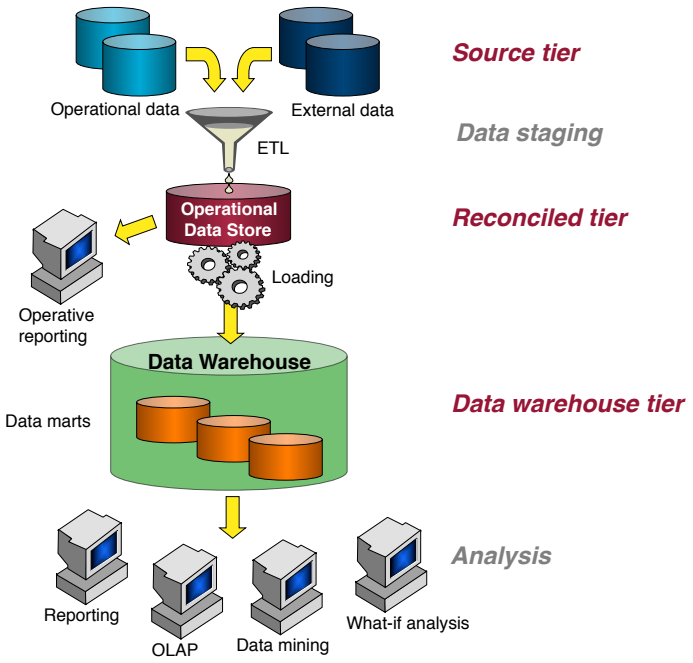
- Introduction & Motivation
- Prescriptive approaches to DW design
- Is *agile* the answer?
- Four Wheel Drive
- The role of conceptual modeling
- The role of CASE tools
- The role of testing

Introduction & Motivation

What is a data warehouse?

- A *data warehouse* is a repository of information aimed at supporting the decisional process. It is:
  - ✓ Subject-oriented
  - ✓ Integrated and consistent
  - ✓ Representing temporal evolution
  - ✓ Non-volatile

Hub-and-spoke architecture



EXTRACTION, TRANSFORMATION, AND LOADING:

ETL processes extract data from sources, transform and clean them, and finally load them in the ODS and in the data warehouse

OPERATIONAL DATA STORE:

Normalized, integrated, consistent, current, and detailed data obtained after integrating and cleansing source data

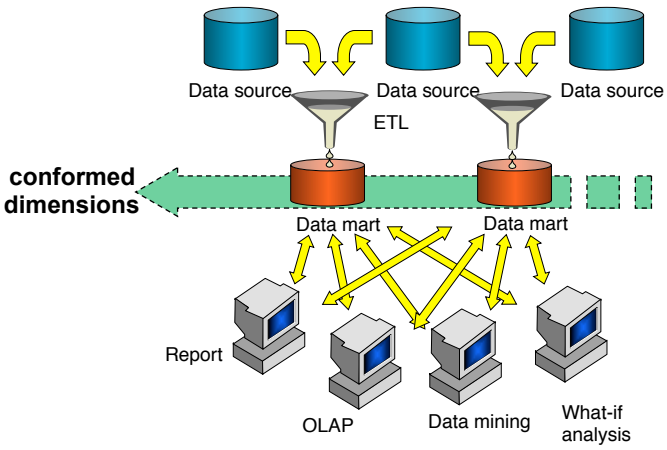
DATA WAREHOUSE:

Multidimensional (often denormalized), integrated, consistent, historical, and aggregate data

DATA MART:

A subset or an aggregation of the data stored into the data warehouse. It includes a set of information pieces relevant to a specific business area, corporate department, or category of users

Data mart bus architecture



CONFORMED DIMENSIONS:

relevant business concepts shared by most data marts

Nice, but...

- Data warehouse systems are characterized by a **long** and **expensive** development process that hardly meets the ambitious requirements of today market
- This causes a **low penetration** of data warehouse systems in small-medium firms
- Data warehouse projects sometimes leave both **customers** and **developers dissatisfied**

And why is this?

- The available literature on data warehouse design mainly focuses on **traditional, linear approaches** (waterfall) that...
  - ✓ ...have a **lose relationship with the sophisticated design methodologies** delivered by the software engineering community
  - ✓ ...yield **low delivery frequencies**
  - ✓ ...**do not involve business users** to a sufficient degree to encourage role-based BI
- Some works appeared about agile data warehousing, but there are evidences that **applying an agile approach *tout court* to data warehouse design has several risks**

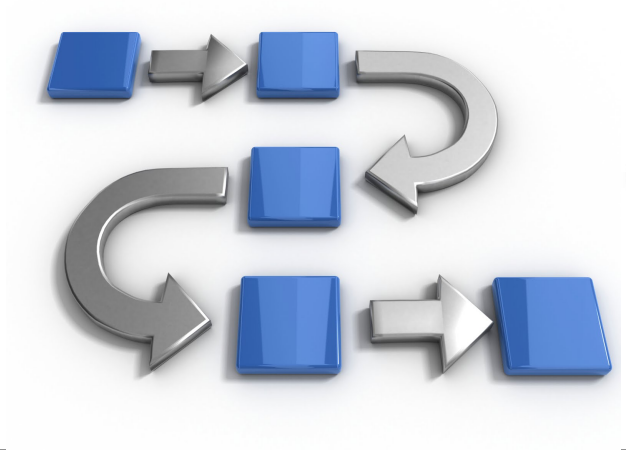
Roles in BI

- User profiling
  - ✓ In operational applications, each profile enables different functions
  - ✓ In DW applications, OLAP ensures broad functional coverage; profiling is essentially used to grant/restrict access to data, and profiles are **statically** modeled (e.g., using use case diagrams)
- Self-service BI
  - ✓ The idea is that a business user can create cubes and reports (almost) on-the-fly **on the corporate DW**, with no need for ICT people to intervene in the process

Roles in BI

- Situational BI
  - ✓ **Situational data** have a narrow focus on a specific domain problem and, often, a short lifespan for a small group of decision makers with a unique set of needs
  - ✓ *Fusion cubes* approach
    - A fusion cube can be dynamically extended both in its schema and its instances; each piece of data in a fusion cube is associated with a set of annotations that describe its quality, the source it was taken from, its freshness and estimated validity, its reliability, etc.
  - ✓ *Exploratory OLAP*
    - Based on a multidimensional schema of the OLAP cube expressed in RDF vocabularies; starting from this, the system can query data sources, extract and aggregate data, and build a cube
  - ✓ *Schema-on-read* approaches
    - The idea is that a business user or a data scientist can write queries on-the-fly on external unstructured or semi-structured data (for which a multidimensional form is not known), with no need for ICT people to intervene in the process
    - This gives rise to **dynamic analysis roles**, because the multidimensional form given to data is situationally determined according to the specific user's role and task

Prescriptive approaches to DW design



Why?

- Designing a data warehouse is a long and complex project that does not always meet the needs of business users
  - ✓ It is often perceived as too rigid and IT-centric and has become more complex with big data
  - ✓ It requires an accurate planning aimed at devising satisfactory answers to organizational and architectural questions
- The risk of getting an unsatisfactory result in data warehousing projects is particularly high because of high user expectations
  - ✓ Risks related to project management
  - ✓ Risks associated with technology
  - ✓ Risks related to data and design
  - ✓ Risks related to the organization
- **Methodologies** are created by closely studying similar experiences and minimizing the risks for failure by founding new approaches on a constructive analysis of the mistakes made previously

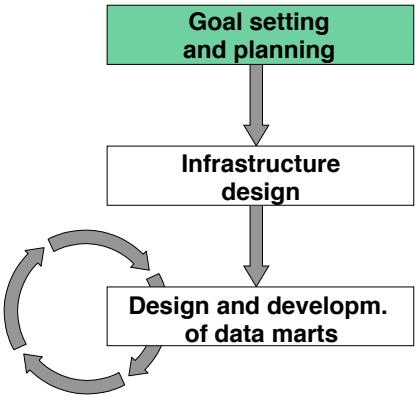
Top-down approach

- Analyze global business needs, plan how to develop a data warehouse, desi whole
  - This procedu lobal picture of the goal to achi istent, well integrated d
  - High-cost ns discourage company r d of projects
  - Analyzing sources at the same time is a very difficu s not very likely that they are all available and stable a time
  - It is extremely difficul ast the specific needs of every department involved oject, which can result in the analysis process coming to dstill
  - Since no working em is going to be delivered in the short term, users cannot chec or this project to be useful, so they lose trust and interest in it

Bottom-up approach

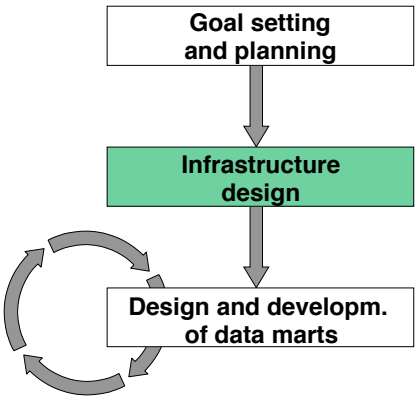
- DWs are incrementa built and several data marts are iteratively created. E data mart is based on a set of facts that are linked to a speci artment and that can be interesting for a user group
  - Leads to concrete res rt time
  - Does not
  - Enables d me
  - Gives mar al benefits of the system bei
  - Keeps the high
  - May determi domain

The life-cycle



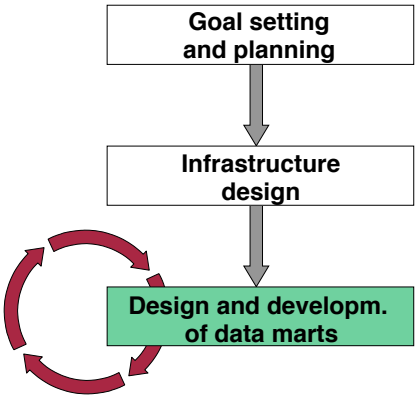
- set system goals, borders, and size
- select an approach for design and implementation
- estimate costs and benefits
- analyze risks and expectations
- examine the skills of the working team

The life-cycle



- analyze and compare the possible architectural solutions
- assess the available technologies and tools
- create a preliminary plan of the whole system

The life-cycle



Every iteration causes a new data mart and new applications to be created and progressively added to the DW system

Data mart design

db administrator

Source Analysis and Integration

Front-end Programming

Requirement Analysis

end-user

Physical Design

Testing

Conceptual Design

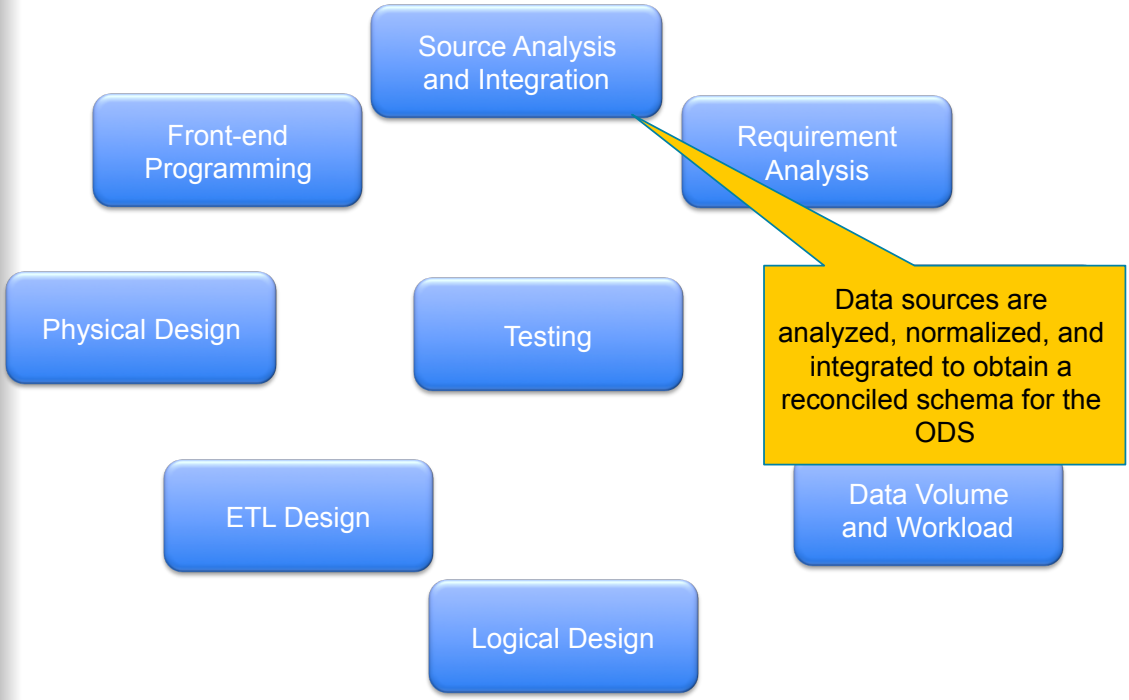
ETL Design

Data Volume and Workload

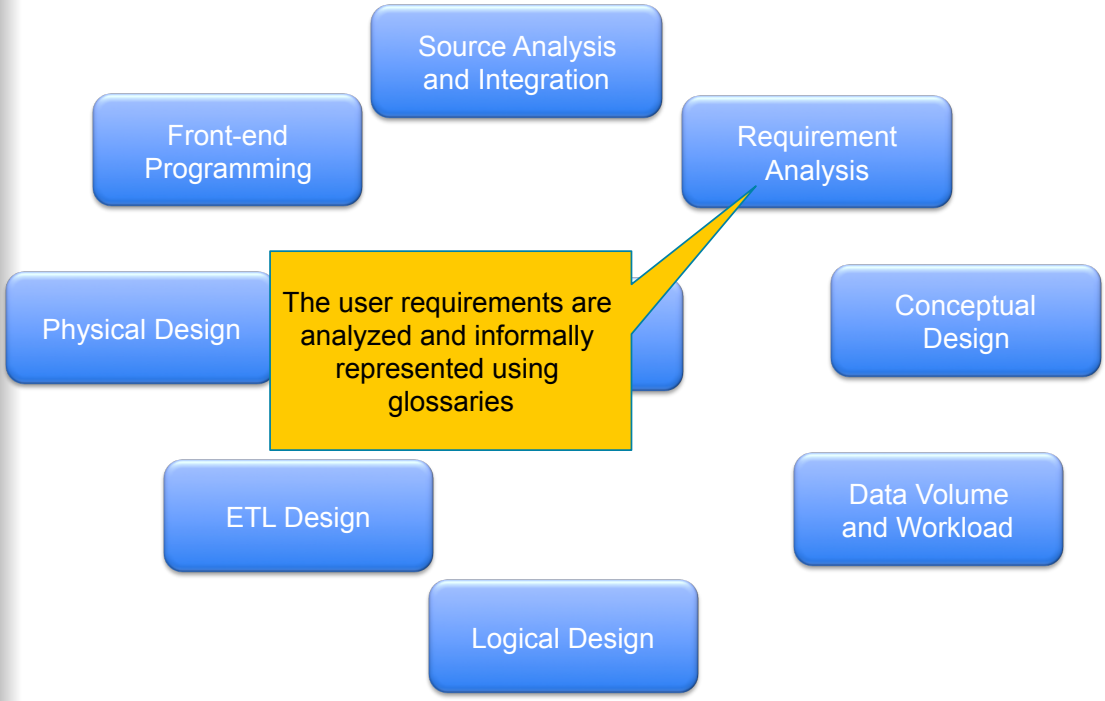
designer

Logical Design

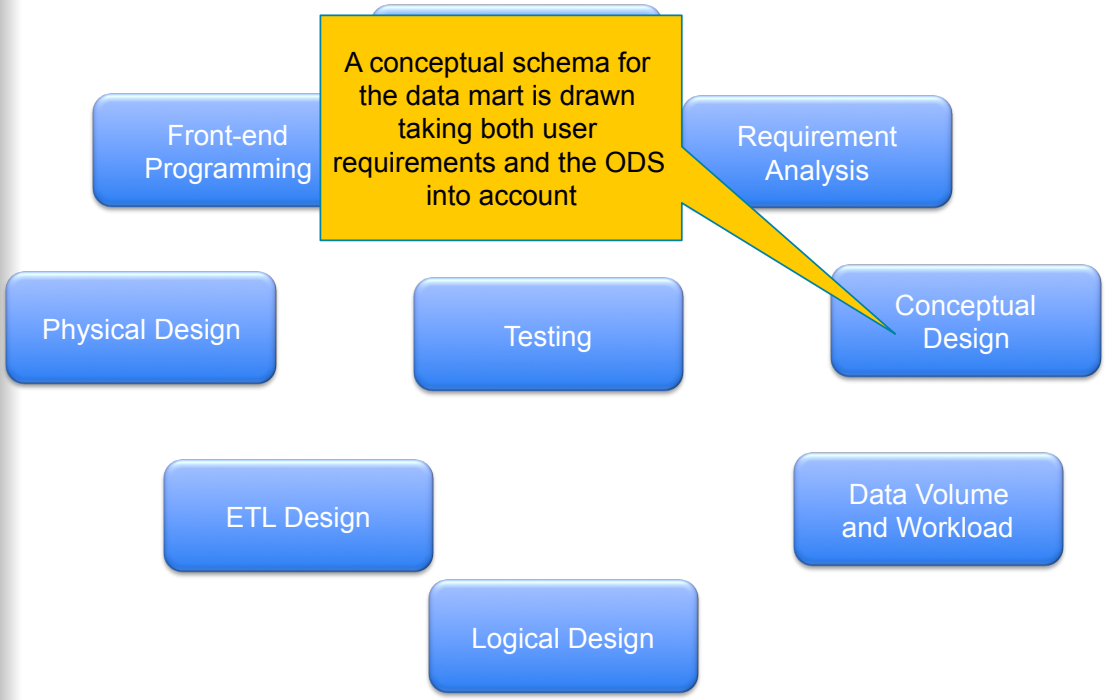
Data mart design



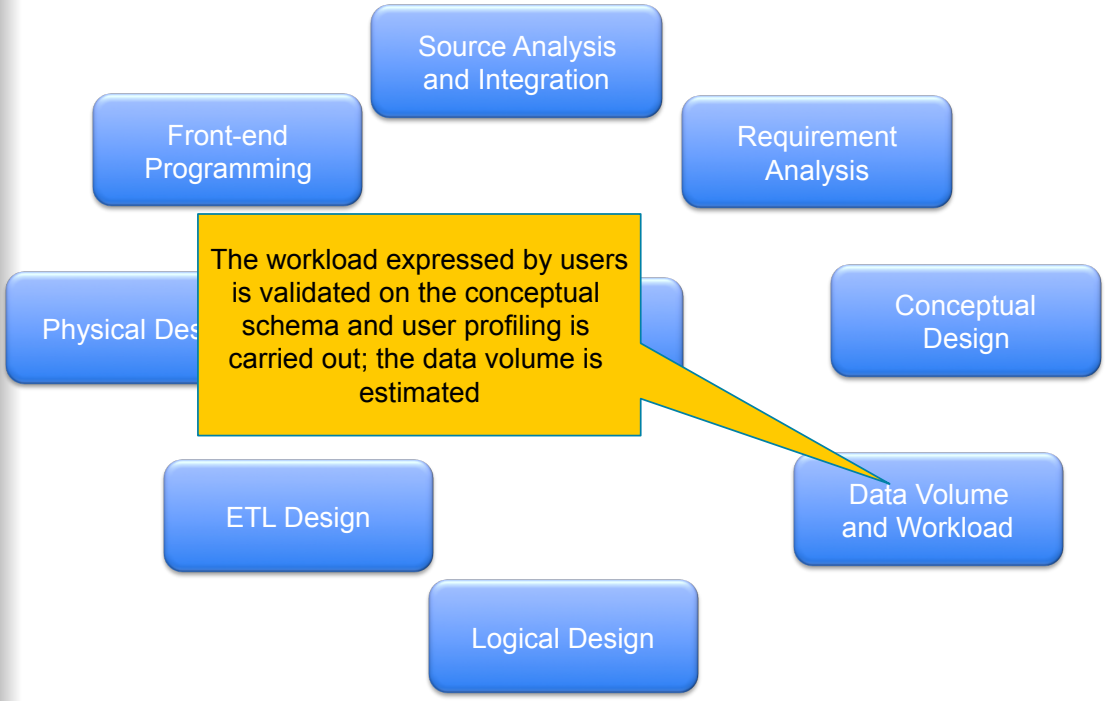
Data mart design



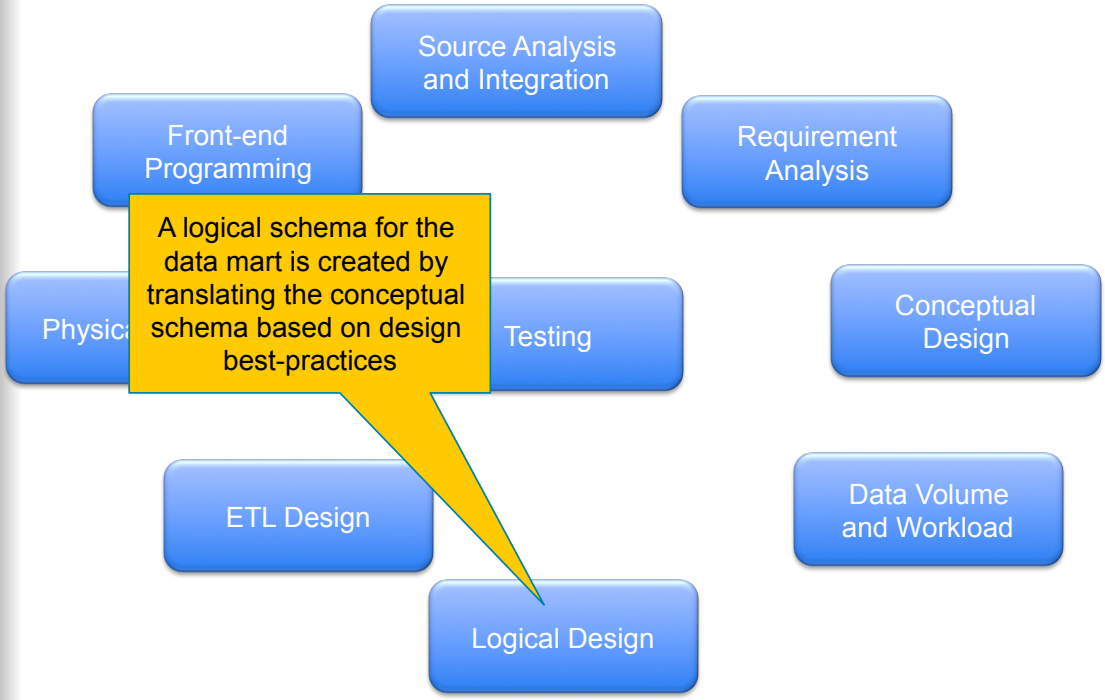
Data mart design



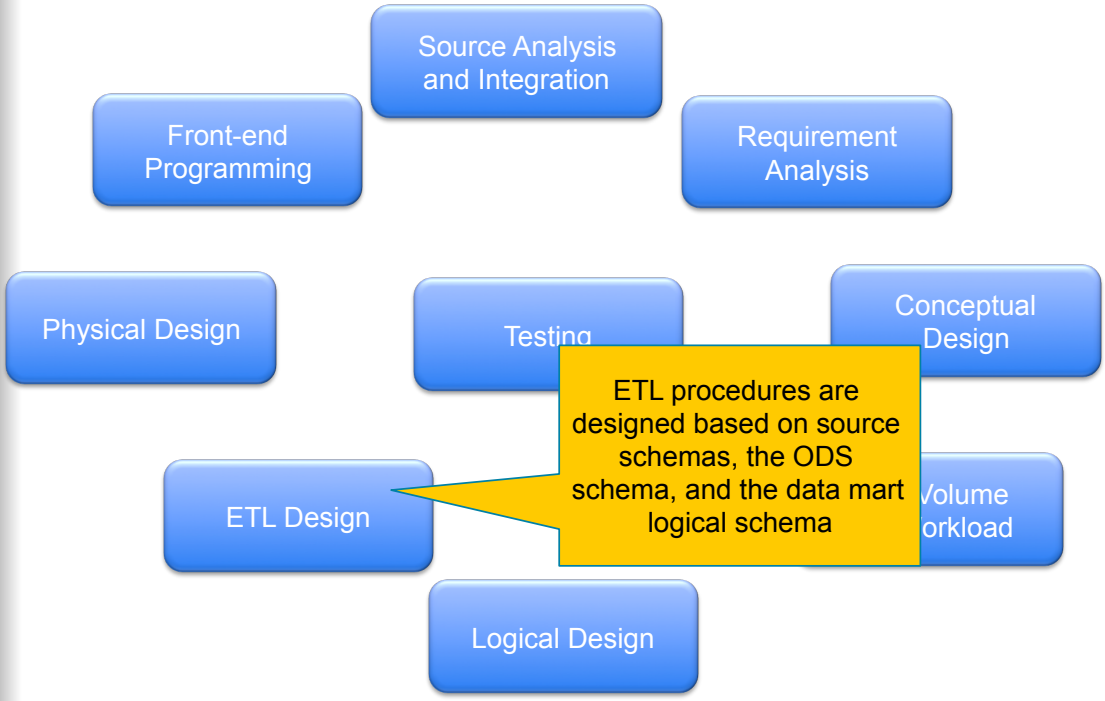
Data mart design



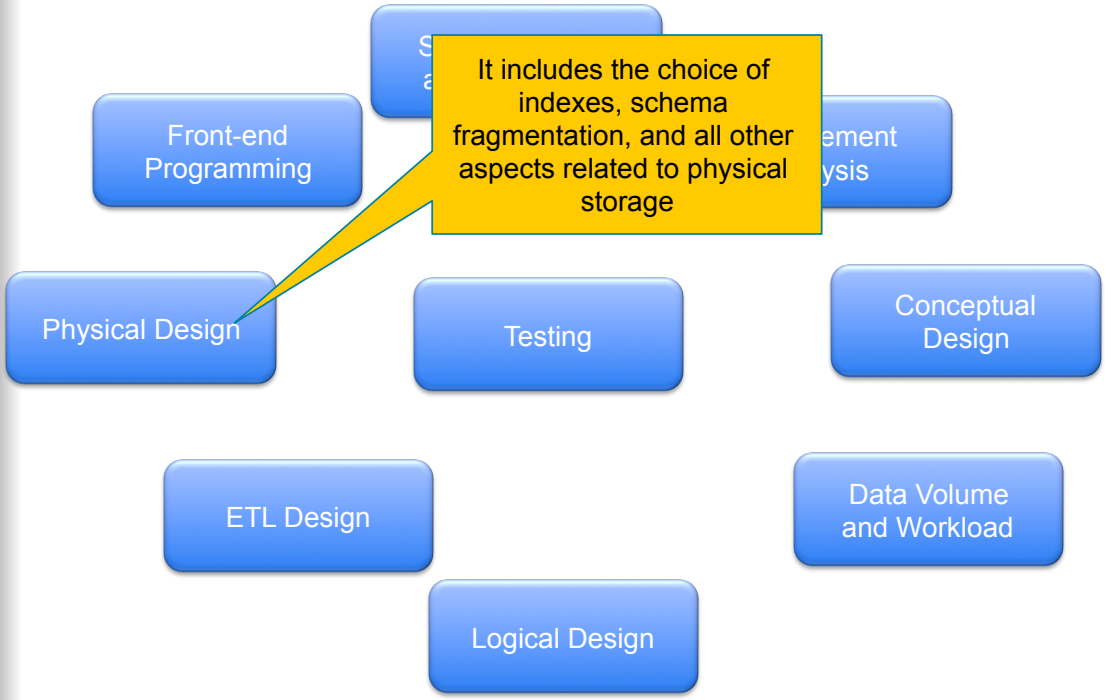
Data mart design



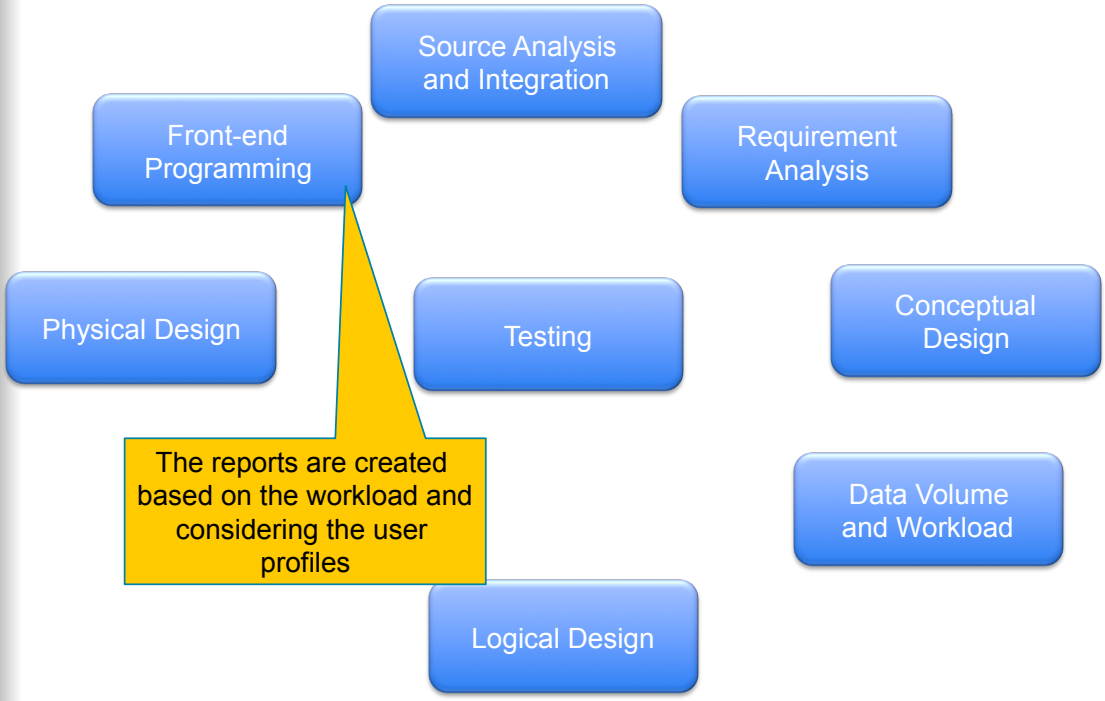
Data mart design



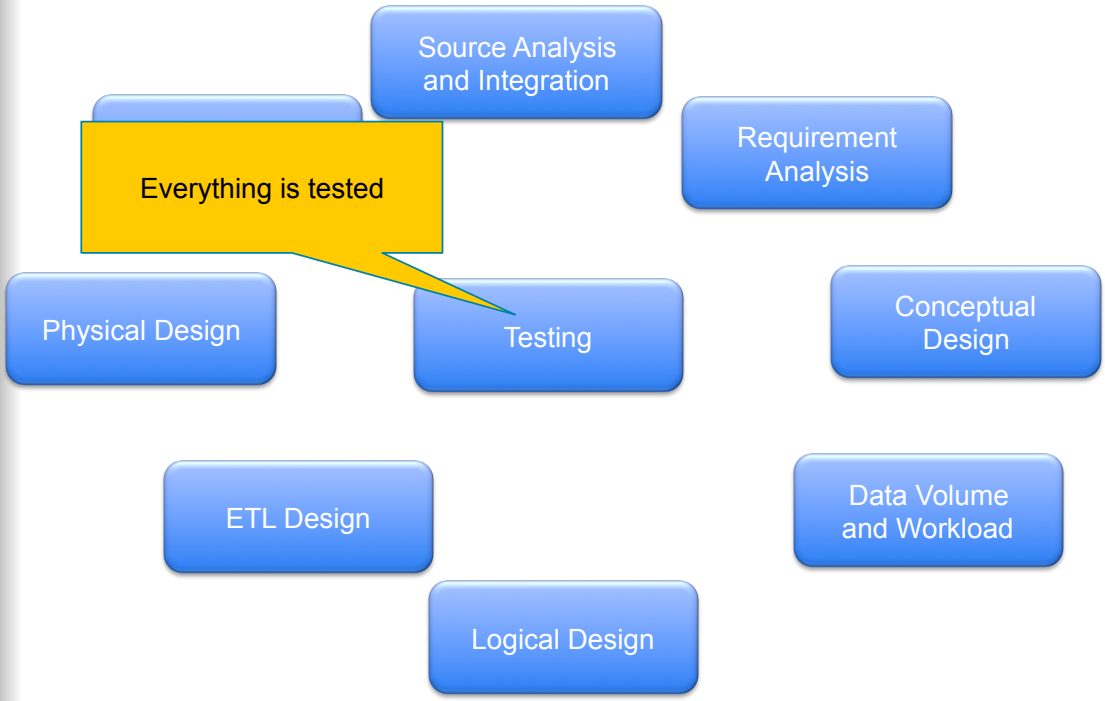
Data mart design



Data mart design



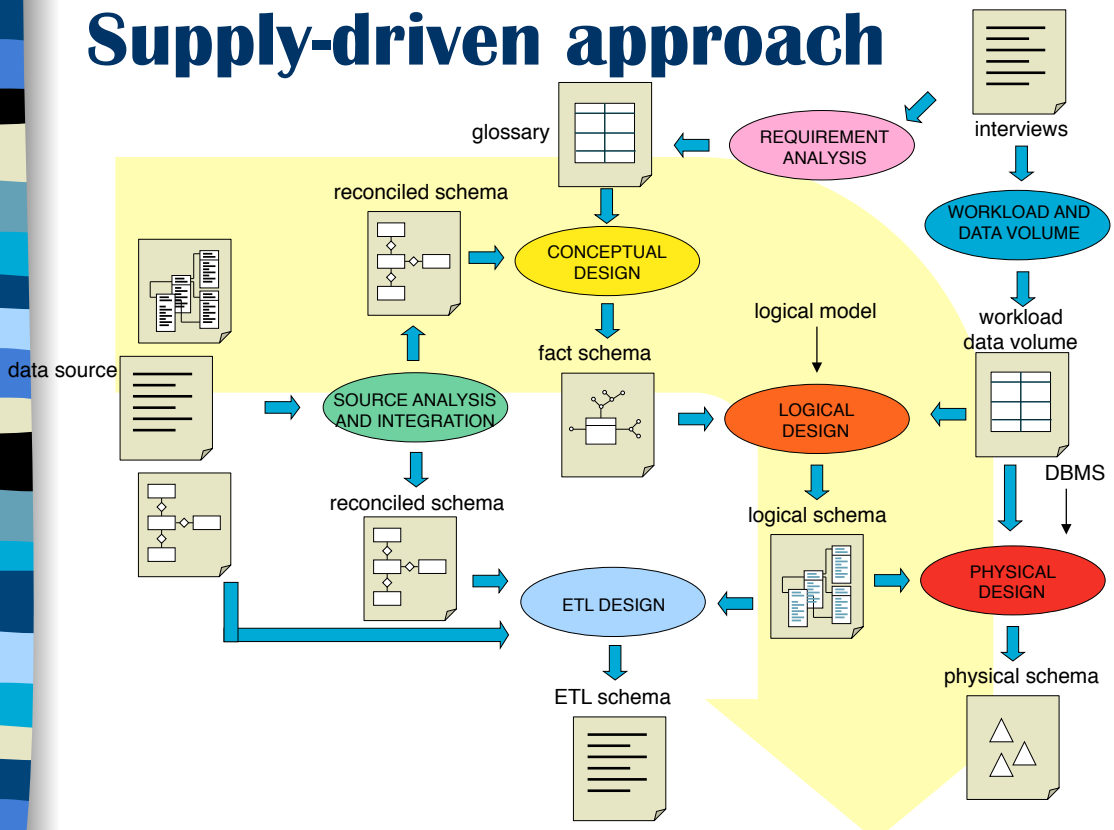
Data mart design



Methodological scenarios

- *Supply-driven approach*
  - ✓ data marts are designed based on a close analysis of operational data sources
  - ✓ user requirements show designers which groups of data, relevant for decision-making processes, should be selected and how to define data group structures based on the multidimensional model
- *Demand-driven approach*
  - ✓ it begins with the definition of information requirements of data mart users
  - ✓ the problem of how to map those requirements into existing data sources is addressed at a later stage, when ETL procedures are implemented

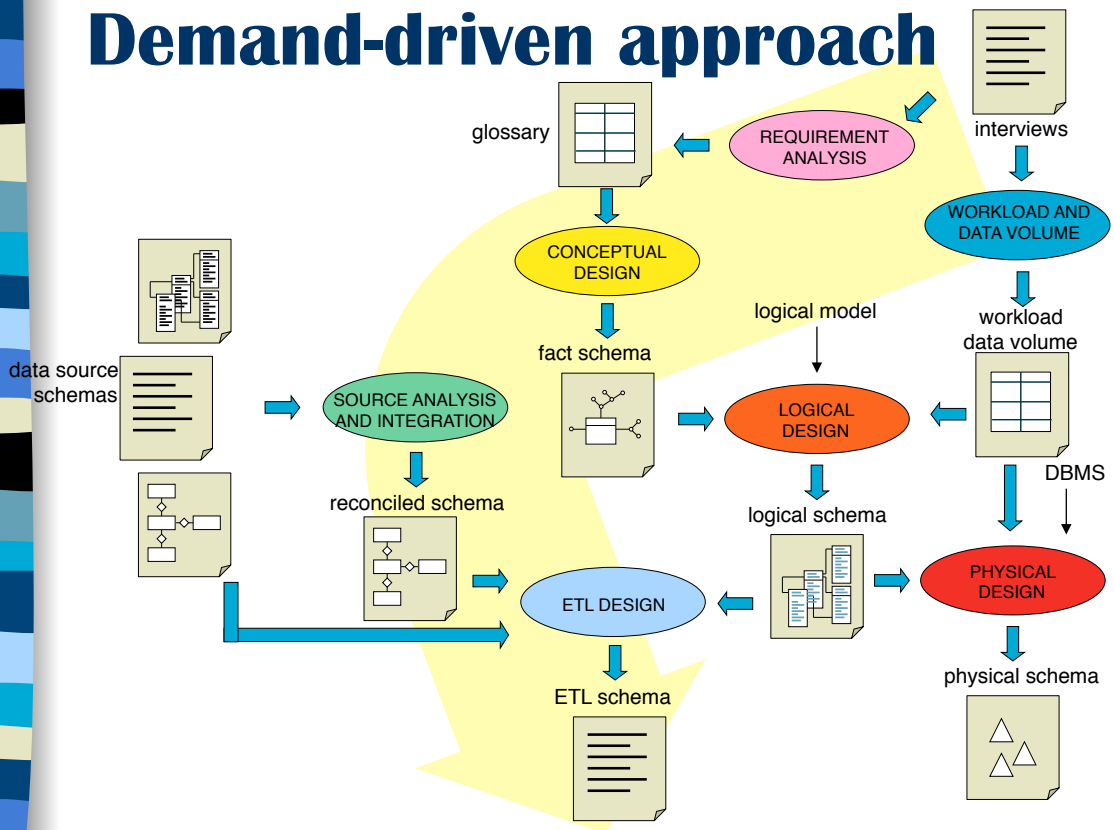
Supply-driven approach



Supply-driven approach

- Pros
  - ✓ an initial conceptual schema for data marts can be automatically derived from the data sources
  - ✓ ETL design is extremely streamlined because every single information piece stored in a data mart is directly associated with one or more source attributes
  - ✓ the resulting data marts are quite stable in time, because they are rooted in source schemas—that change less frequently than the requirements expressed by end users
- Cons
  - ✓ it can only be applied when there is enough a priori knowledge of the data sources

Demand-driven approach



Demand-driven approach

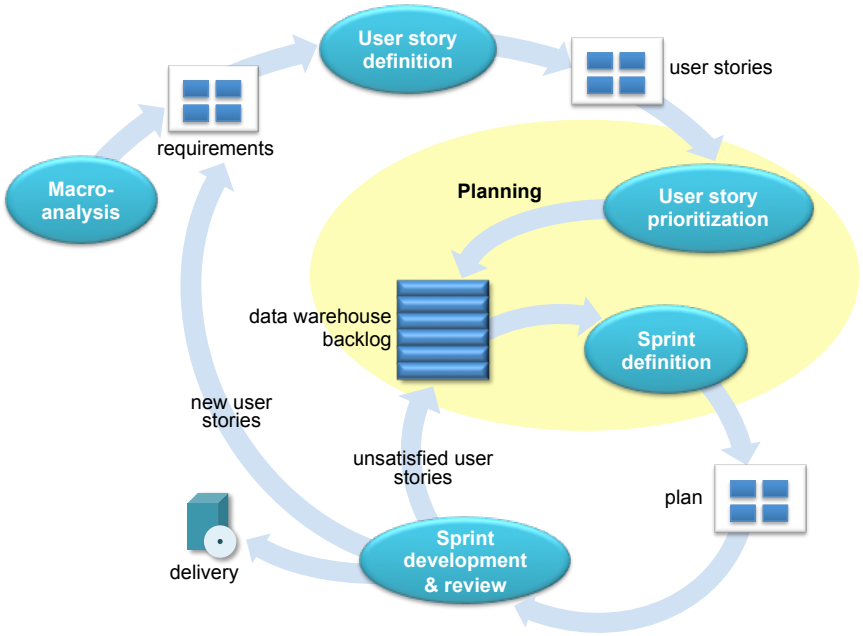
- Pros
  - ✓ users' wishes play a leading role
- Cons
  - ✓ designers are required to have strong leadership and meeting facilitation qualities to properly grab and integrate the different points of view
  - ✓ designers make great efforts in the data-staging design phase
  - ✓ facts, measures, and hierarchies are drawn directly from the specifications provided by users, and only at a later stage can designers check for the information required to be actually available in source databases

Is agility the answer?

An agile view

- Agile methods, such as *Scrum* and *eXtreme Programming*, are based on the twelve principles stated in the Agile Manifesto; they are non-prescriptive product development methods in which there is no prescribed sequence
- Key practices:
  - ✓ Incremental and iterative design and implementation: the software is described in terms of detailed user functionalities (*user stories*), and at each iteration (usually 2 to 6 weeks) the team should deliver the set of user stories that maximizes the utility for the users while fulfilling a set of constraints
  - ✓ Team awareness: iteration planning is based on sharing and averaging the estimates given by team members about story complexity, utility, non-delivery risk, and dependencies
  - ✓ User-centered design and user involvement: continuous interaction with users is promoted to progressively refine the specifications, reduce inadequate requirements, and increase the trust between users and developers
  - ✓ Continuous and automated testing: to facilitate requirement validation and obtain better results, the system is developed by refining and expanding an evolutionary prototype that progressively integrates the implementation of each increment
  - ✓ Lean documentation: a well-defined documentation is a key feature to comply with user requirements. Small and simple formal schemas are preferred to extensive specifications

An agile view



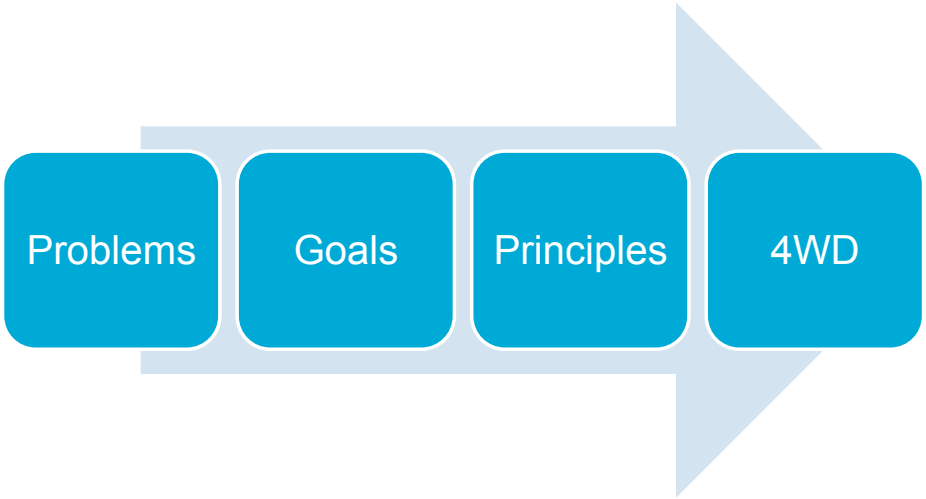
An agile view

- Agile methodologies can be applied to data warehouse projects only partially
  - ✓ The link between agile and DW is tenuous
    - DW platforms consist of software in an analytics use-case, while agile is used to develop software
  - ✓ Highlight on incremental delivery of valuable components for users...
    - ...but several DW components are hardly perceived as valuable by users
  - ✓ Strong project segmentation based on user stories: high-level functional requirements that can be implemented in a few days...
    - ...but founding design on detailed functionalities required by users does not allow the multidimensional structure of data to be correctly determined
  - ✓ Agile asks, "what data is needed right now and what are the sources we will address?"
    - This may lead the designer to do exclusively bottom up modeling, i.e., addresses column from the data sources as if they were the “only” data and then rationalize those columns almost “as-is” into a single model

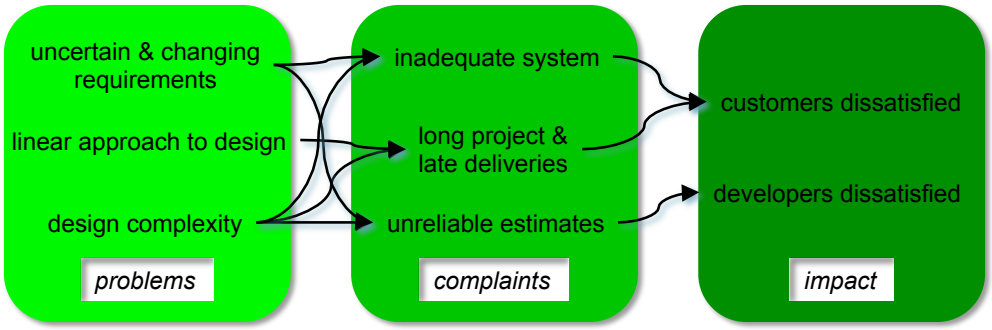
Four Wheel Drive

Modus operandi

- Analyze the potential advantages arising from the application of modern software engineering methodologies to a data warehouse project



From problems...



...to goals...

Quality (of the development process)	*Description*	*Effect*
Reliability	Probability that the delivered system completely and accurately meets user requirements	ensure high-quality and satisfactory final system
Robustness	process flexibility, i.e., capability of quickly and smoothly reacting to unanticipated changes in the environment	better manage uncertain and changing requirements
Productivity	Efficiency in using the resources assigned to the project to speed up system delivery	make shorter and cheaper projects
Timeliness	Accuracy of time and cost predictions	make resource estimates more reliable

...to principles...

Principles / Methodologies	Waterfall	Rapid Application Development	Prototyping-Oriented Software Development	Spiral Software Development	Model-Driven Architecture	Component-Based Software Engineering	Agile Software Development
Incrementality and risk-based iteration		✓		✓			
Prototyping			✓				
User involvement		✓					✓
Component reuse						✓	
Formal and light documentation	✓				✓		✓
Automated schema transformation					✓		

...to principles...

Principles / Goals	Reliability	Robustness	Productivity	Timeliness
Incrementality and risk-based iteration	Continuous feedback, clearer requirements	Better management of change	Better management of project resources, rapid feedback	Early detection of errors
Prototyping	Frequent tests, easier error detection		Early deliveries	
User involvement	Better requirement validation, better data quality			Early detection of errors
Component reuse	Error-free components		Faster design	Predictable development
Formal and light documentation	Clearer requirements	Easier evolution	Faster design	
Automated schema transformation	Optimized performances	Easier evolution	Faster design	Predictable design

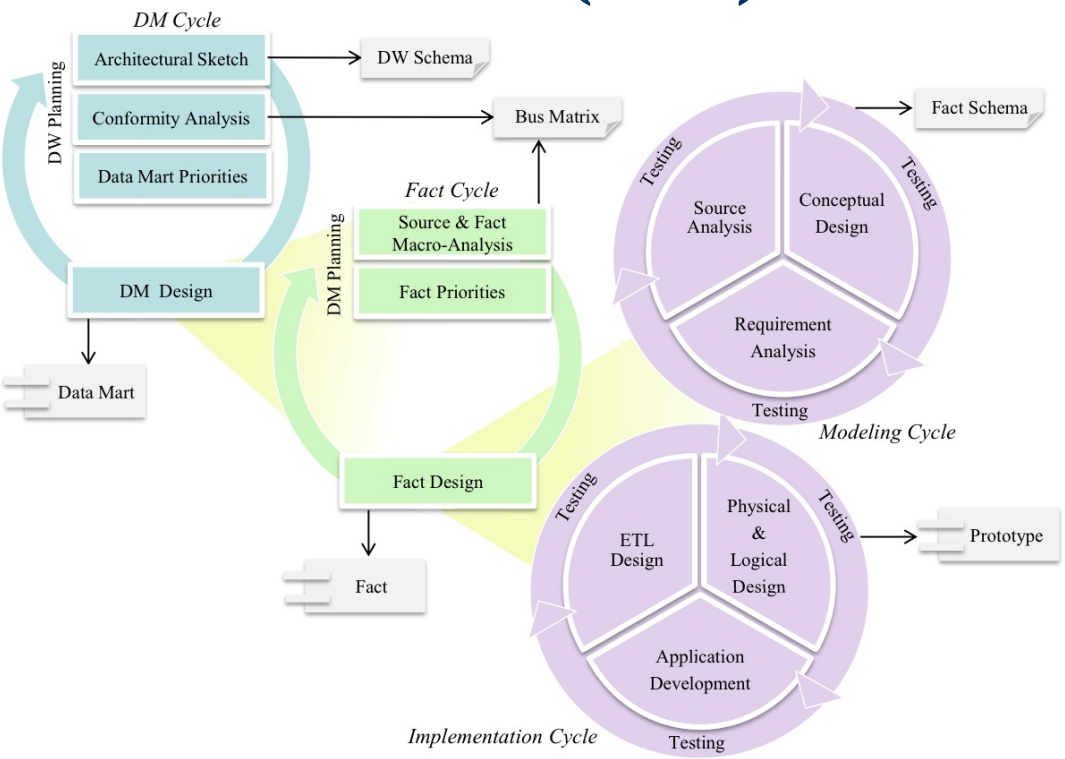
...to methodology

- A methodology arising from the application of software engineering principles to data warehouse projects
  - ✓ Fact-based iteration cycles
  - ✓ Build a conceptual schema before implementation
  - ✓ User involvement in testing activities
  - ✓ Possibility of changing fact and data mart priorities

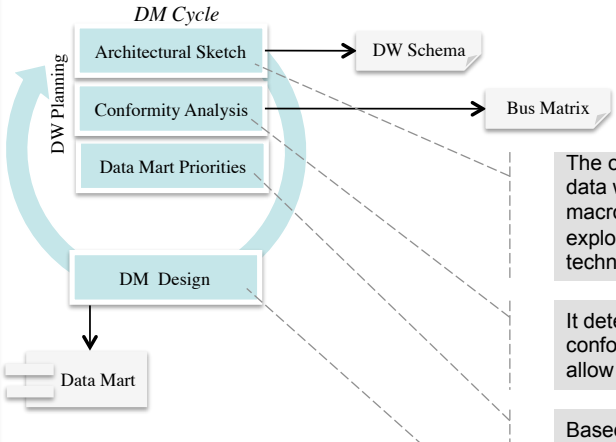
Four-Wheel-Drive (4WD)

- Nested iteration cycles:
  - ✓ **Data mart cycle**
    - defines and maintains a global plan for the development of the whole data warehouse
    - incrementally designs and releases data marts
  - ✓ **Fact cycle**
    - refines the data mart plan
    - incrementally designs and releases the facts of a data mart
  - ✓ **Modeling & Implementation cycles**
    - include the analysis, design, and implementation activities for delivering reports and applications concerning a single fact

Four-Wheel-Drive (4WD)



4WD: Data mart cycle



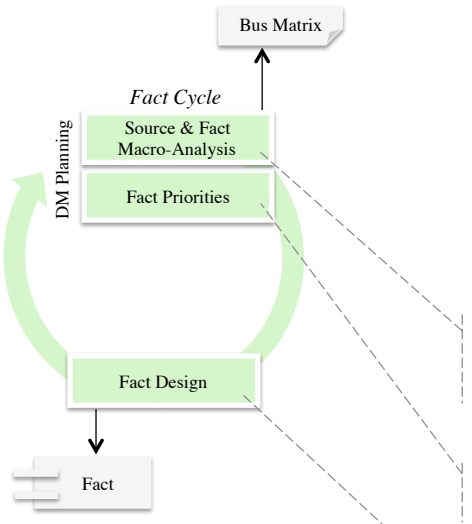
The overall functional and physical architecture of the data warehouse is progressively drawn based on a macro-analysis of user requirements and an exploration of data sources as well as on budget, technological, and organizational constraints

It determines which dimension of analysis will be conformed across different facts and data marts, to allow cross-fact analysis and obtain consistent results

Based on a trade-off between user priorities and technical constraints

It builds and releases the top-priority data mart. After each data mart has been built, the three phases above are iterated to allow the data warehouse plan to be refined and updated

4WD: Fact cycle

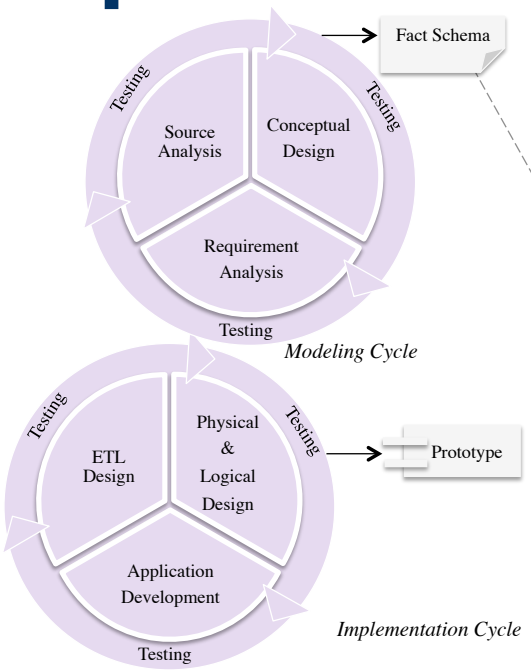


Aimed at checking the availability, quality, and completeness of the data sources and determining the main business facts to be analyzed by users

It is the result of a trade-off between user requirements and technical priorities

It develops and releases the top-priority fact. After that, the two phases above are iterated to allow the data mart plan to be refined and updated

4WD: Modeling & implementation cycle

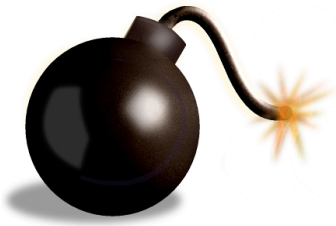


The fact schema takes a leading role in separating the modeling and the implementation activities

How 4WD supports...

- Incrementality
  - ✓ Functional requirements in DW projects are mainly expressed in terms of analysis capabilities, so agile DW design often focuses each iteration on a small set of reporting or OLAP functionalities
    - this can lead to increasing the design effort, because it gives little relevance to the the design of multidimensional schemas
    - functionality-centered iterations fail in recognizing that apparently different analyses, designed during separate iterations, are actually supported by the same multidimensional schema
  - ✓ In 4WD, the shortest iterations that release a tangible result to users are those for modeling and implementing a single fact (2-4 weeks overall)
    - the modeling and implementation cycles actually have a daily to weekly frequency
    - the deliveries they produce enable a progressive refinement of the conceptual schema and implementation through a massive test based on active involvement of users

How 4WD supports...



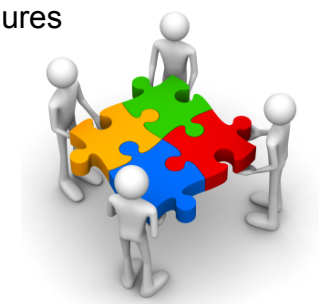
- Risk-based iterations
  - ✓ Risk guides the data mart and fact priority definition
  - ✓ At the data mart level
    - Give priority to data marts that include widely shared hierarchies
    - Give priority to data marts that are fed from stable and well-understood data source
    - Postpone data marts based on unclear requirements, assuming that these requirement will be better understood as the user's involvement in the project increases
  - ✓ At the fact level
    - Give priority to facts that include the main business hierarchies and require the most complex ETL procedures
    - Adopt a data-driven approach to design rather than a requirement-driven one whenever users do not appear to have a deep knowledge of the business domain
    - Plan the length of an iteration in proportion to the complexity of the fact, since failing a release in the early stage of a project will undermine the team credibility

How 4WD supports...

- Prototyping
  - ✓ Prefer an *evolutionary* (a robust prototype is continuously refined) and *incremental* (the prototype is gradually enlarged by adding new sub-systems or pursuing new qualities) approach to a *throw-away* approach (the prototype is used to demonstrate a small set of functions and then is abandoned)
    - the effectiveness of prototyping is maximized when the prototype is tested together with users, which in a DW project requires the whole data flow to be prototyped: an effort that should not be wasted
  - ✓ Use prototyping to support every DW development phase:
    - To help designers validate requirements
    - To improve the design of reports and analysis applications
    - To advance testing to the early phases of design
    - To evaluate the feasibility of alternative solutions during logical design and ETL design

How 4WD supports...

- User involvement
  - ✓ Tight collaboration between users and designers
    - Preliminary user training (e.g., to clarify project goals, introduce a shared terminology, explain the multidimensional model)
    - Prototyping to favor user awareness of the project status
    - User feedback to detect problems and errors
      - For usability tests of reporting and OLAP front-ends
      - For functional tests of ETL procedures



How 4WD supports...

- Component reuse
  - ✓ Favor the use of predefined elements to support the data warehouse development
    - Conformed hierarchies
    - Library hierarchies (for a given domain)
    - Library facts (for a given domain)
    - ETL building blocks
    - Analysis templates

How 4WD supports...

- Formal and light documentation
  - ✓ Formal but lean documentation to formalize requirements, simplify communication, support accurate design, and encourage maintainability
  - ✓ At the data warehouse level
    - A high-level schema to summarize data marts, data sources, and user profiles
  - ✓ At the data mart level
    - Bus matrix for conformity analysis
  - ✓ At the fact level
    - Conceptual schema released and validated *before* proceeding with the implementation

role of conceptual modeling

How 4WD supports…

- Automated schema transformation
  - ✓ Automate design steps to boost the DW life-cycle
    - Supply-driven conceptual design
    - Reverse engineering
    - Logical design based on best-practices

The role of conceptual modeling

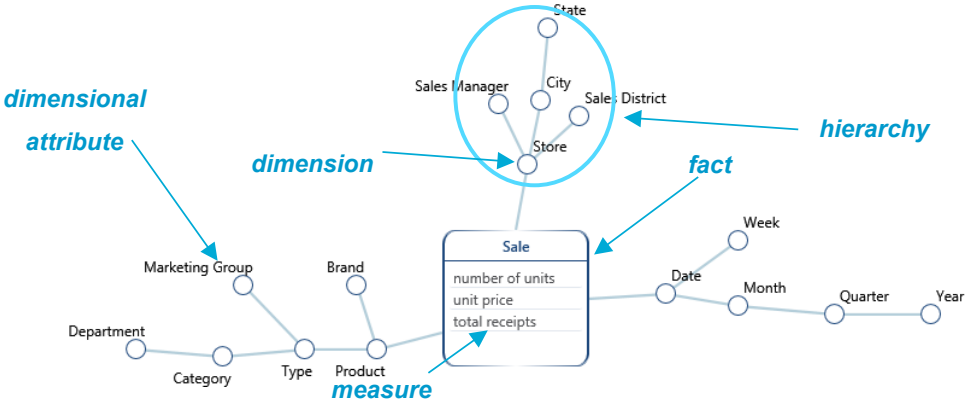
Which formalism?

- While it is now universally recognized that a data mart is based on a multidimensional view of data, there is still no agreement on how to implement its conceptual design
  - ✓ Use of the Entity-Relationship model is quite widespread throughout companies as a conceptual tool for standard documentation and design of relational databases, but *it cannot be used to model DWs*
  - ✓ Designers often base their data marts design on the logical level—that is, they directly define star schemas. But a star schema is nothing but a denormalized relational schema; *it contains only the definition of a set of relations and integrity constraints!*

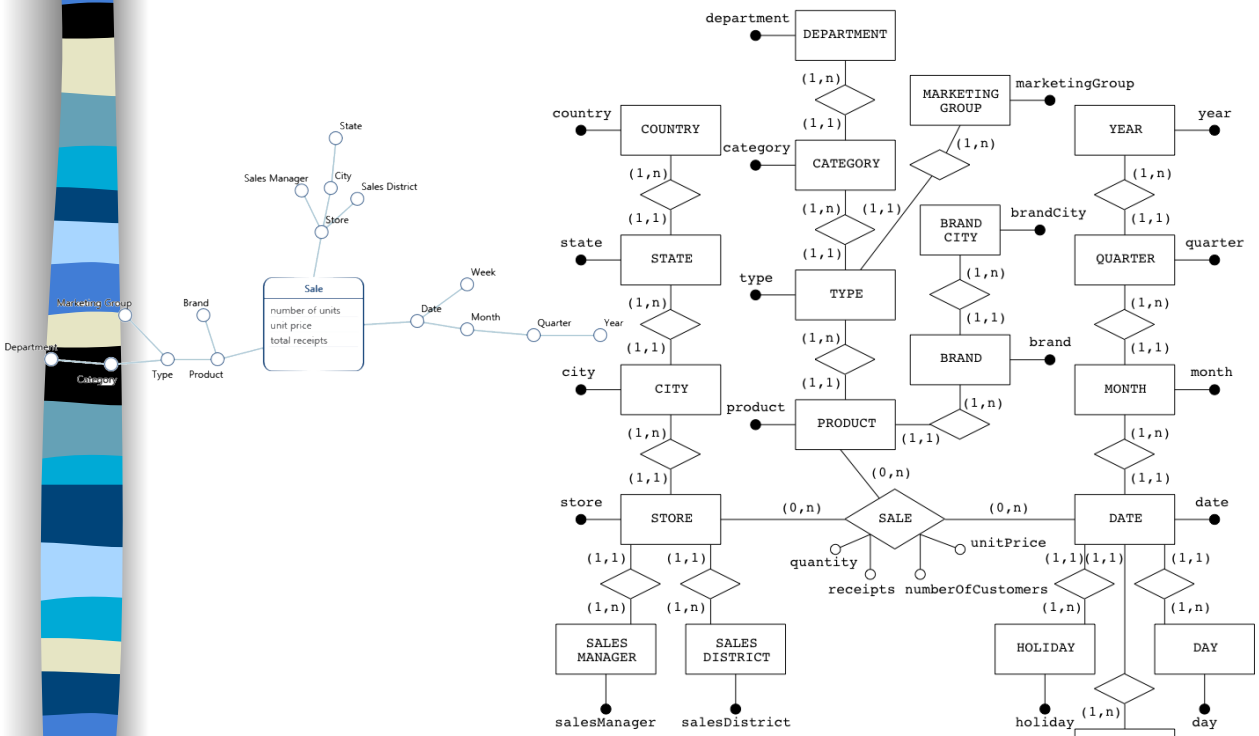
The Dimensional Fact Model

- The DFM is a graphical model devised to:
  1. lend effective support to conceptual design of data marts
     - it is implementation-independent
     - it is expressive
     - it is non-ambiguous
     - it is formally sound (based on FD theory)
     - it can be automatically translated into a logical schema
  2. be easily understood by both designers and end-users
  3. provide clear and expressive project documentation
- It has been successfully experimented over the last 20 years in both the academic and industrial worlds
- The conceptual representation generated by the DFM consists of a set of *fact schemas* that model facts, measures, dimensions, and hierarchies

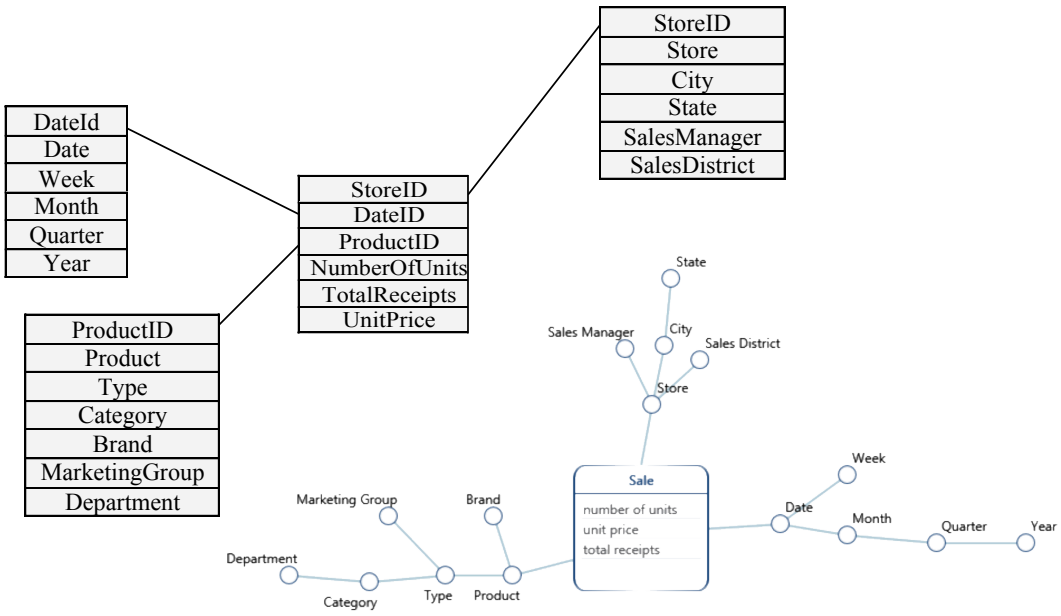
DFM: basic expressiveness



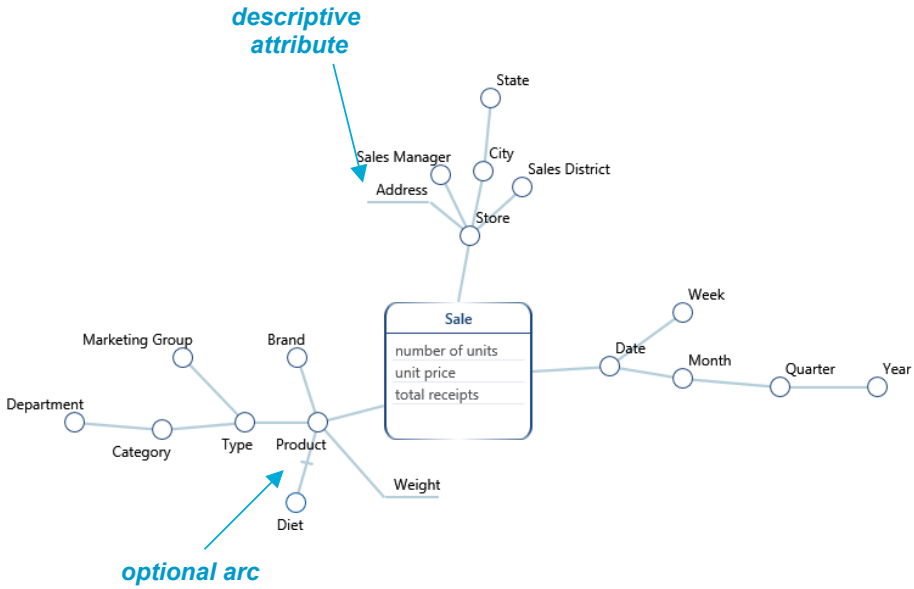
DFM vs. ERM



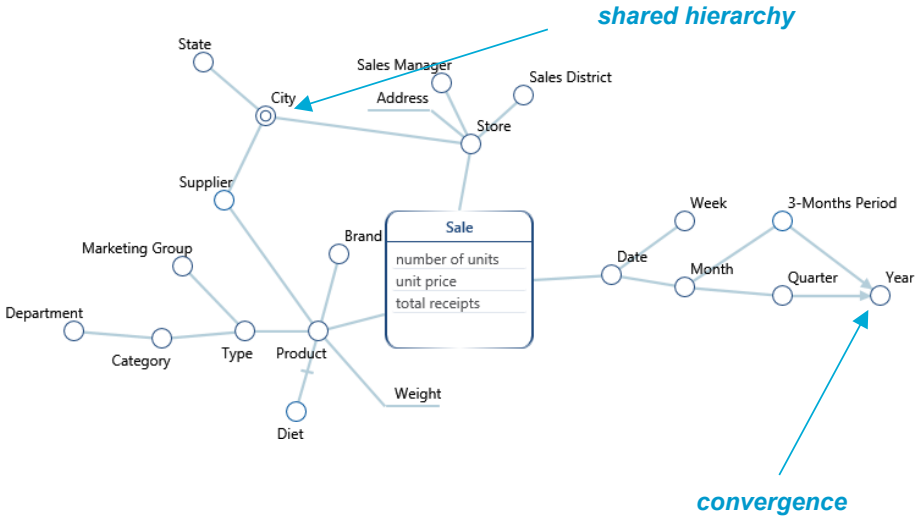
DFM vs. star schema



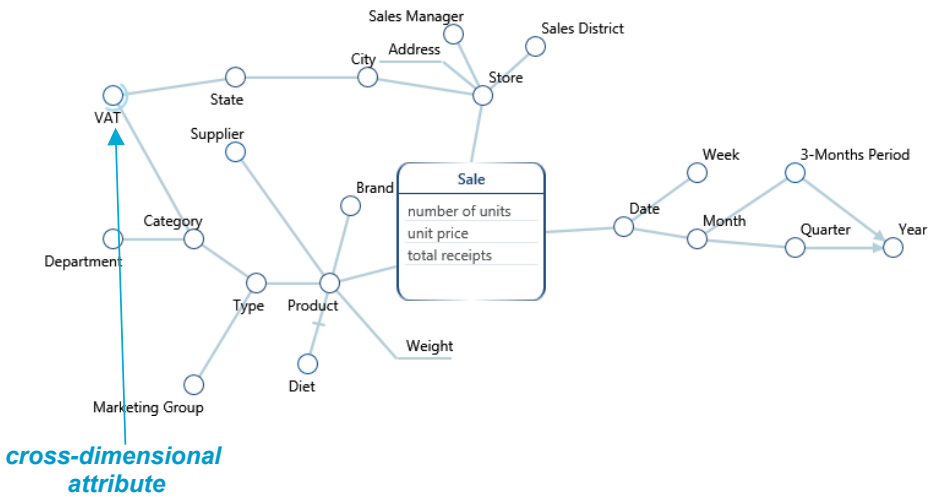
DFM: advanced expressiveness



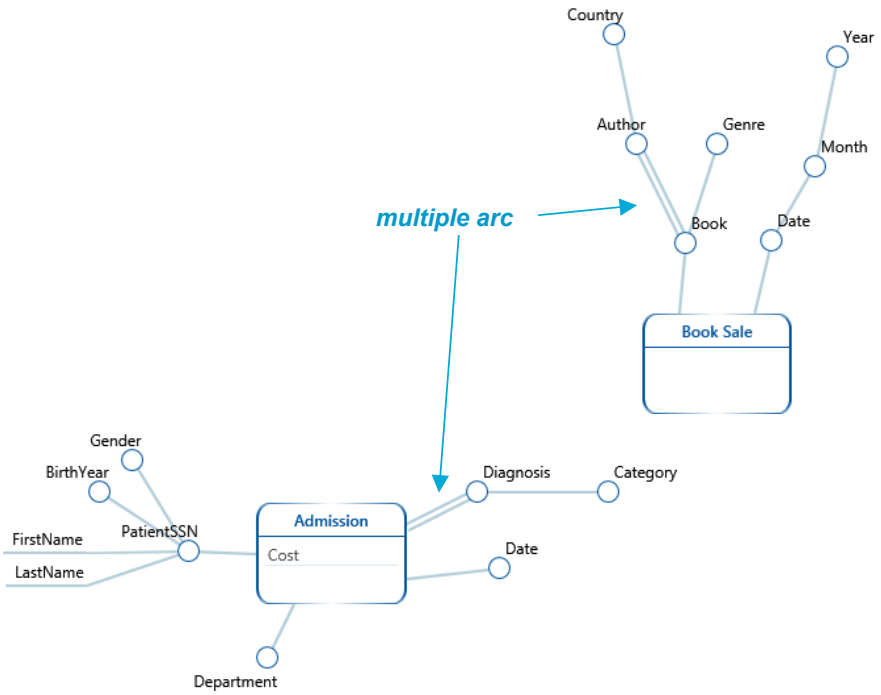
DFM: advanced expressiveness



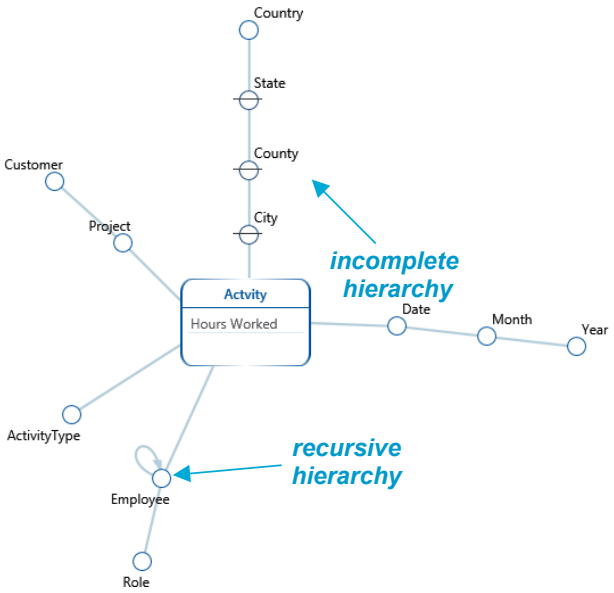
DFM: advanced expressiveness



DFM: advanced expressiveness



DFM: advanced expressiveness



DFM: advanced concepts

Sale

	Product	Stores	Date
number of units	Sum	Sum	Sum
unit price	Avg	Avg	Avg
total receipts	Sum	Sum	Sum

additivity matrix

Advantages of the DFM

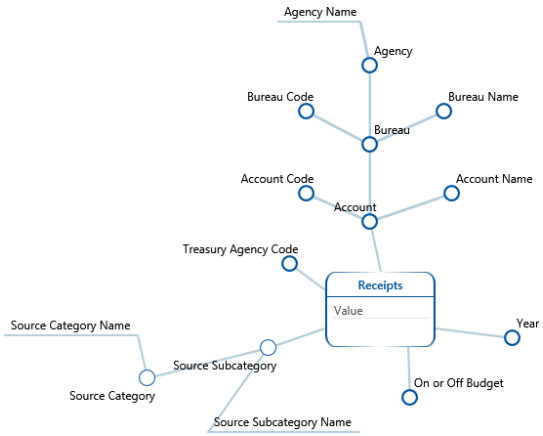
1. it gives designers and end-users a platform-independent, non-ambiguous, comprehensive picture of the DW content
2. it is 100% independent of the OLAP multidimensional engine chosen for deployment and of the target logical model
3. it enables effective communication between designers and end-users with the goal of more accurately formalizing requirement specifications
4. it decreases the overall complexity of design by breaking it into two distinct but inter-related phases
5. it streamlines the DW life-cycle by enabling logical design to be automated based on widely-recognized best practices

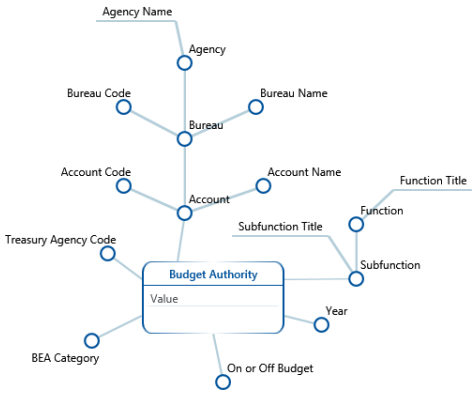
Advantages of the DFM

6. it provides clear and expressive design documentation, readable by both ICT and business people, which improves the overall system maintainability
7. it effectively integrates user-oriented business glossaries
8. it enables early testing of requirements based on the core workload expressed by users, thus reducing the probability of errors and misunderstanding
9. it encourages self-service BI by helping business users understand the information content of their DW
10. it encourages situational BI by enabling users to better understand the multidimensional context they are working with
11. it enables computation of metrics to effectively assess the quality of design
12. it is supported by a CASE tool

The DFM in action

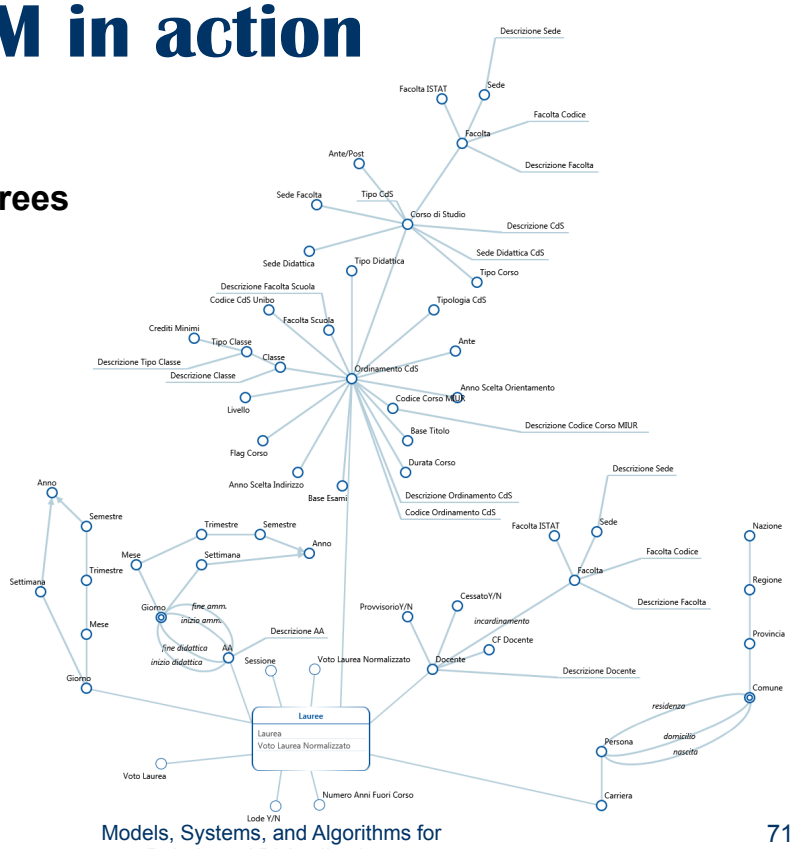
Budget of the United States Government for Fiscal Year 2016





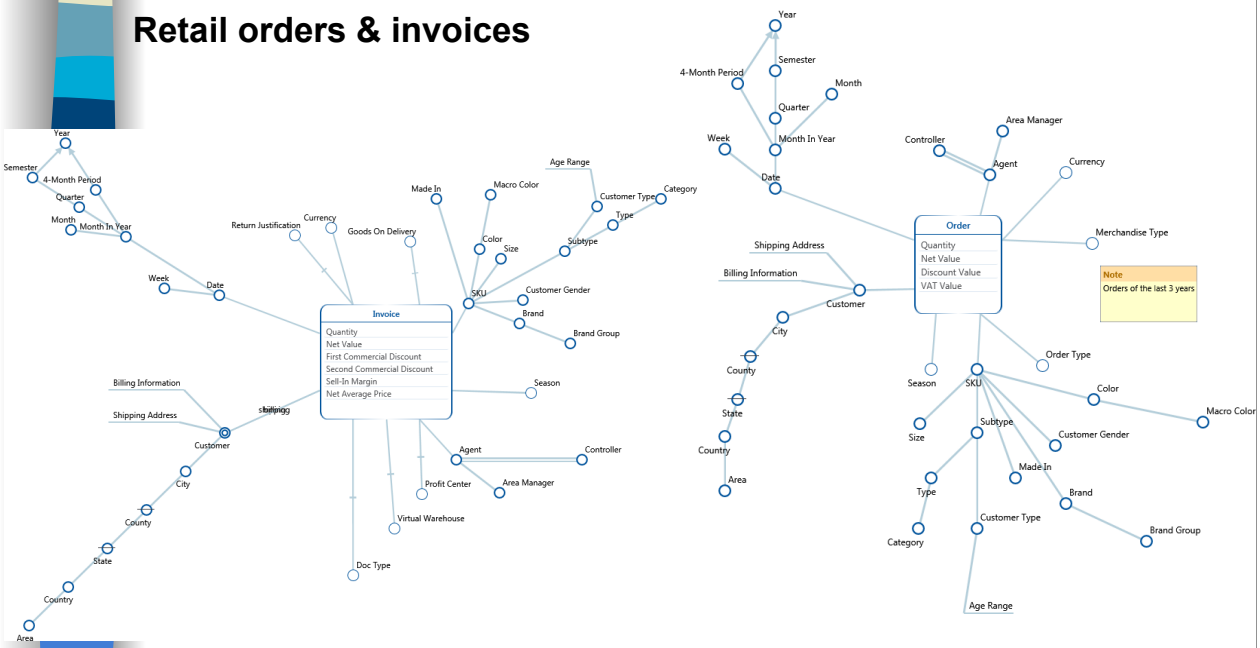
The DFM in action

University degrees



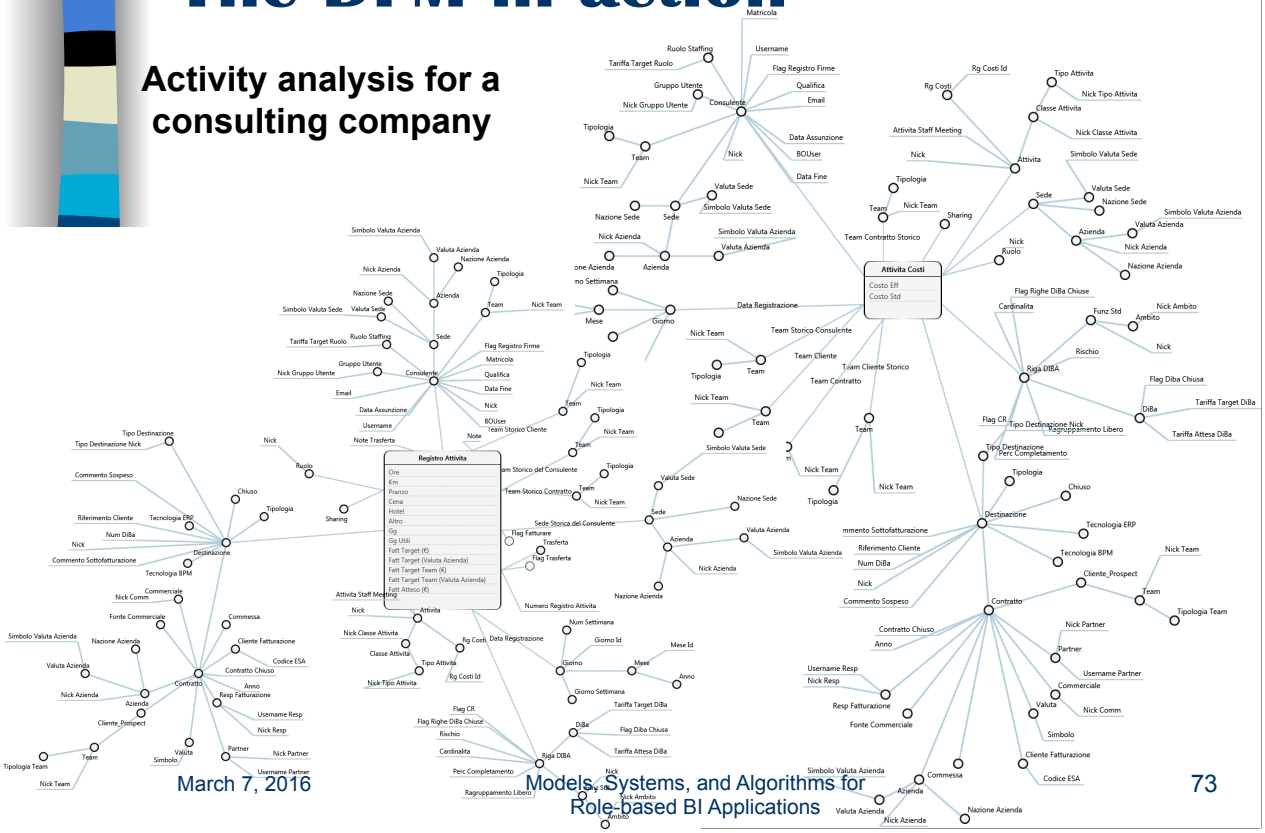
The DFM in action

Retail orders & invoices



The DFM in action

Activity analysis for a consulting company



The role of CASE tools

Benefits of CASE tools

- Speed up design
- Create, update, and manage documentation
- Make system maintenance and evolution easier
- Keep track of project versions
- Enable teamwork

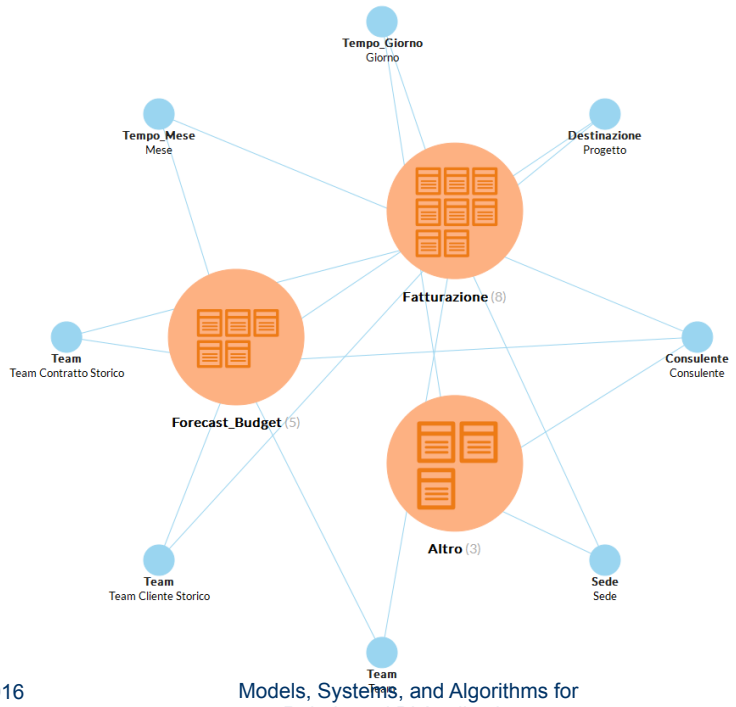
It is well-known among software engineers that devising a design methodology is almost useless, if there is no CASE tool to support it

The indyco Suite

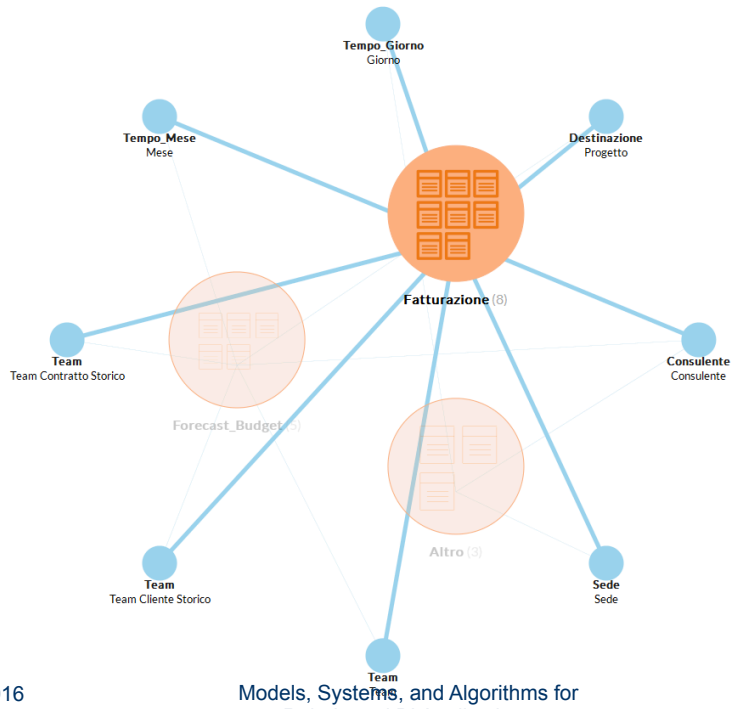
- Demand-driven conceptual design with syntactical validation
- Reverse engineering from star/snowflake schemas
- Logical design based on preferences
- Generation of SQL DDL
- Automated documentation
- Business glossary
- Project quality metrics

- Navigation of multidimensional schemas for business users
- Business glossaries
- Different abstraction levels
- Search for concepts
- Creation of custom views of schemas

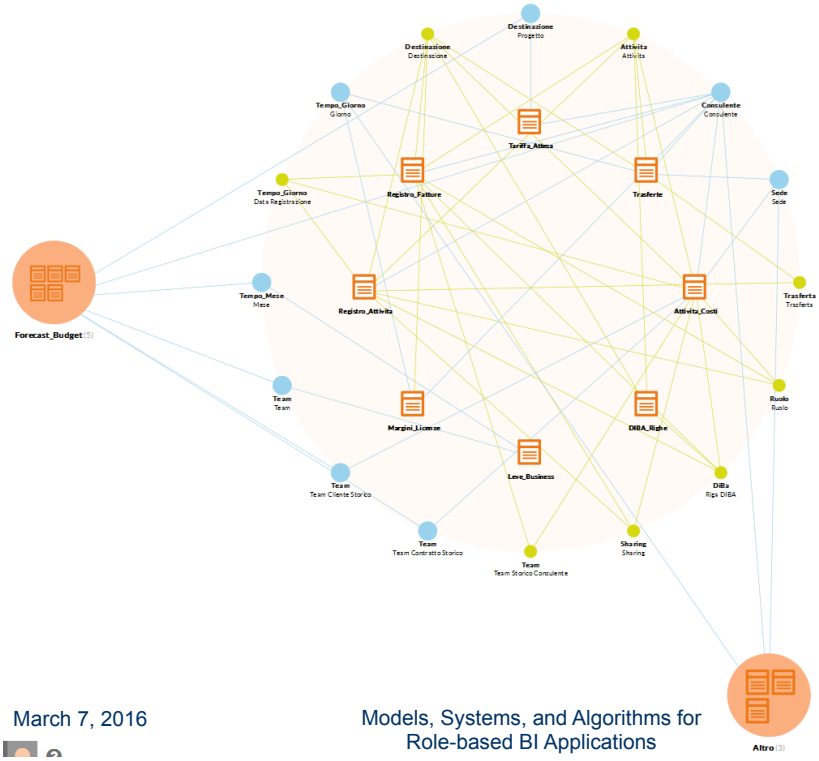
Data warehouse view



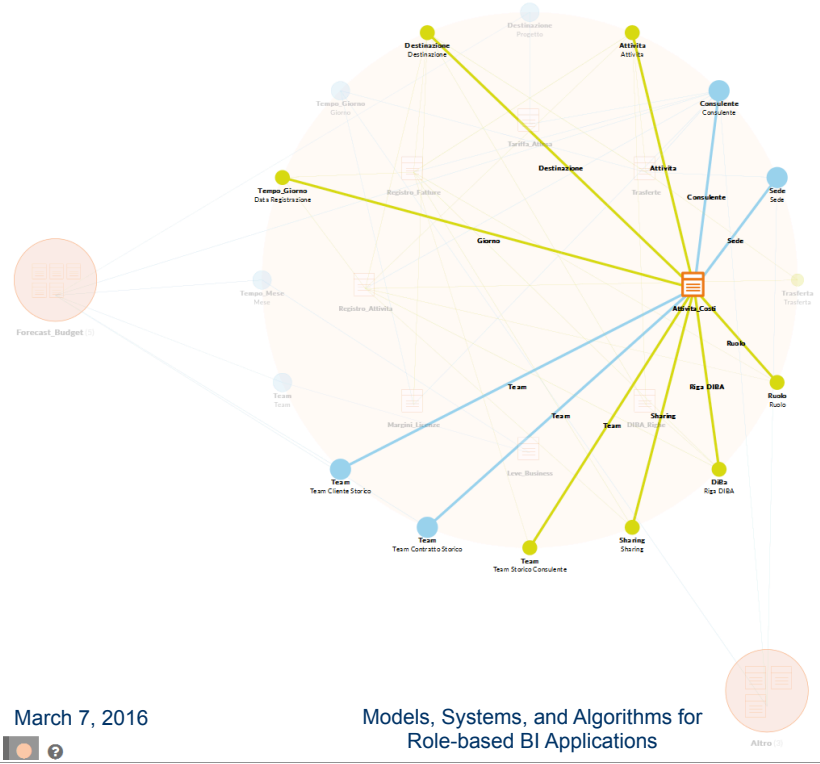
Data warehouse view



Data mart view



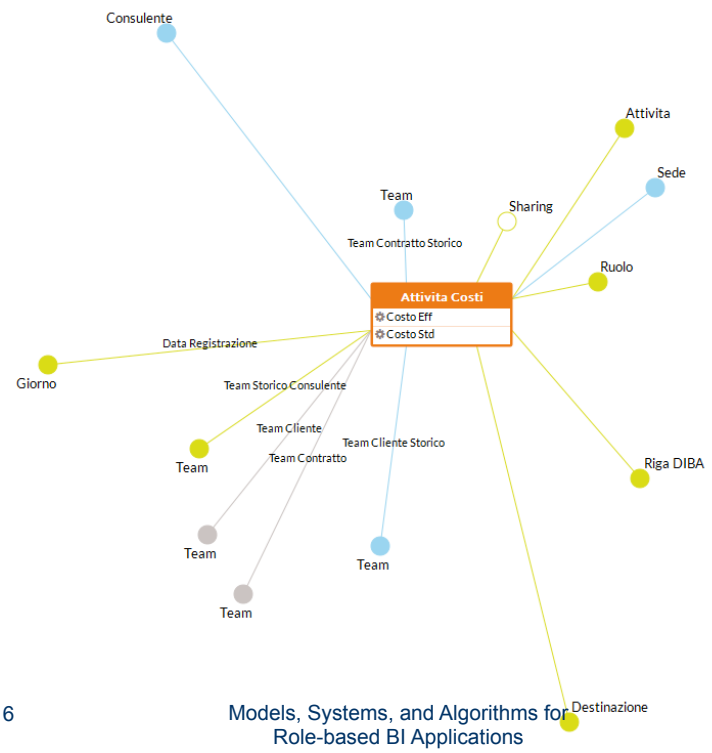
Data mart view



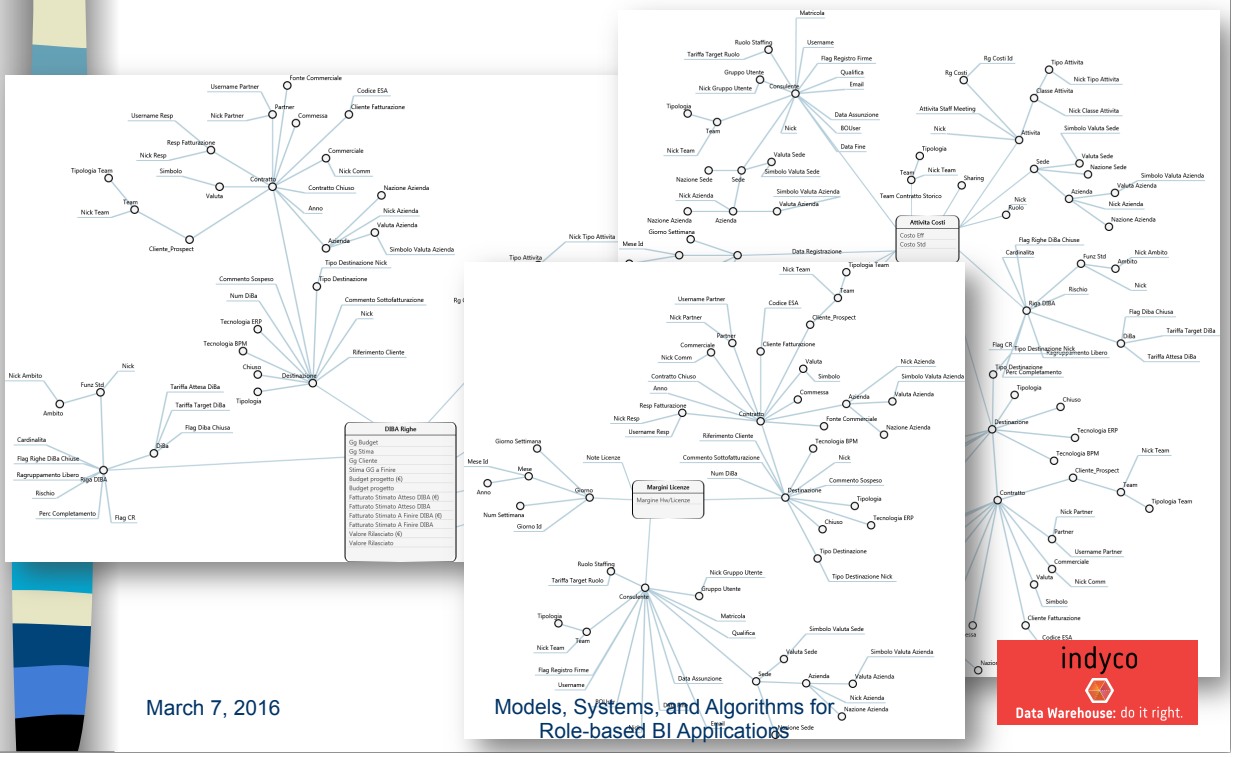
Data mart view

	Attivita_Costi	DIBA_Righe	Leve_Business	Margini_Licenze	Registro_Attivita	Registro_Fatture	Tariffa_Attesa	Trasferte
Attivita Attivita	✔		✔	✔			✔	✔
Consulente Consulente	✔	✔	✔	✔		✔		✔
Tempo_Giorno Data Registrazione	✔		✔	✔				
Destinazione Destinazione	✔		✔	✔		✔	✔	✔
Tempo_Giorno Giorno						✔		✔
Tempo_Mese Mese					✔			
Destinazione Progetto		✔						
DiBa Riga DIBA	✔		✔	✔			✔	
Ruolo Ruolo	✔		✔	✔				
Sede Sede				✔				✔
Sharing Sharing	✔		✔	✔				
Team Team					✔			
Team Team Cliente Storico				✔				
Team Team Contratto Storico				✔				
Team Team Storico Consulente			✔	✔				
Trasferta Trasferta	✔							

Fact view



Fact view



March 7, 2016

Models, Systems, and Algorithms for Role-based BI Applications

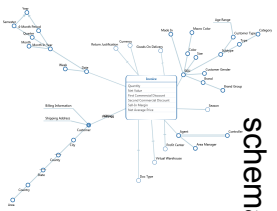
indyco
Data Warehouse: do it right.

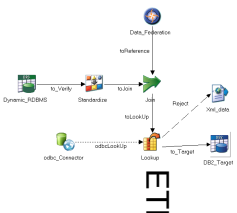
The role of testing

A winning combination

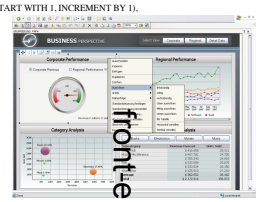
- The earlier an error is detected in the software life-cycle, the cheapest correcting that error is
  - ✓ Multidimensional schemas are typically the first design artifact that is created and can be tested, so they have a primary role in ensuring early discovery of errors
- All DW design methodologies entail a multidimensional modeling phase, that is carried out either at the conceptual level or at the logical level
  - ✓ A conceptual representation is by far more expressive and can more easily be understood by non-expert users
- Adopting a methodology that entails a conceptual design phase has a positive impact on the effectiveness of testing, which in turn brings several advantages in terms of design quality and accuracy and in terms of maintainability and reuse

What vs. how in testing





```
CREATE TABLE DT_DATA
(    chiaveD          INT     NOT NULL
     GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1),
     data             DATE NOT NULL,
     mese             CHAR(6),
     trimestre        CHAR(5),
     anno             CHAR(4),
     giorno           CHAR(10),
     settimana        INT,
     vacanza          CHAR,
     PRIMARY KEY  (chiaveD)   )
CREATE TABLE DT_NEGOZIO
(    ..............
```



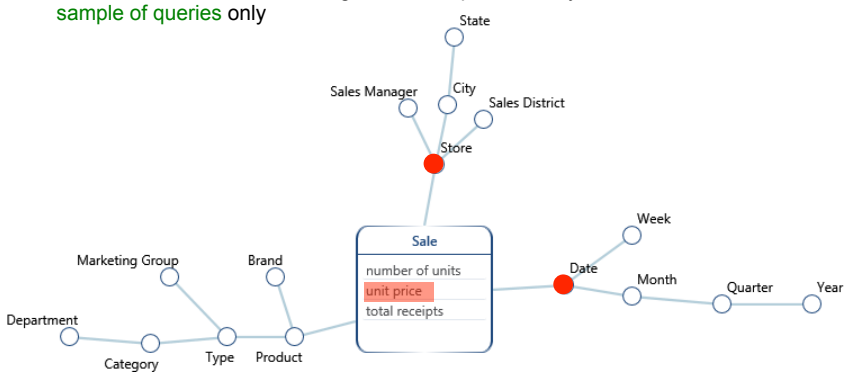
	multidimensional schema	ETL	physical schema	front-end
functional	✓	✓		✓
usability	✓	✓		✓
performance	✓	✓	✓	✓
stress		✓	✓	✓
recovery		✓	✓	
security	✓	✓	✓	✓
maintainability	✓	✓		

Early loading test

- Execute some sample loadings of the data mart starting from source data, using a draft ETL prototype
  - ✓ further check on the correctness of the multidimensional schema
  - ✓ first performance test of the physical schema
  - ✓ reuse components of the prototype for implementing the ETL
  - ✓ better plan a forced-error test of the ETL thanks to evidence of the most common types of errors the ETL will have to handle
  - ✓ functional test of the front-end can be advanced, so misunderstandings and errors in requirements (e.g., an error in the formula for computing a derived measure or a KPI) may be discovered earlier

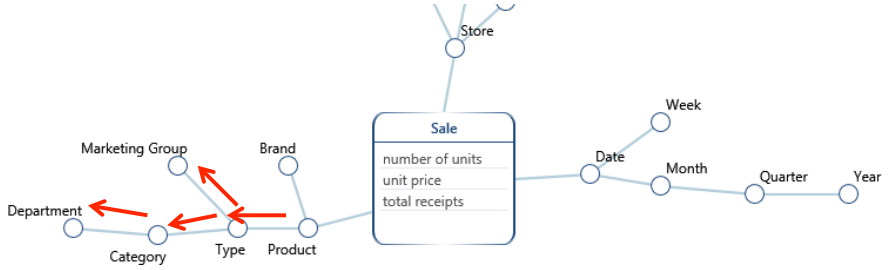
Functional test of DFM schemas

- It aims at verifying that the multidimensional schemas produced for the data mart effectively support user requirements
  - ✓ The **workload test** verifies that the workload preliminarily expressed by users during requirement analysis is actually supported by the multidimensional schema
    - Check, for each workload query, that the required measures have been included in the fact schema and that the required aggregation level can be expressed as a valid grouping set on the fact schema
    - Should the workload be too large to be comprehensively tested, tests can be made on a sample of queries only



Functional test of DFM schemas

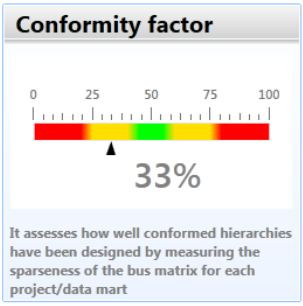
- It aims at verifying that the multidimensional schemas produced for the data mart effectively support user requirements
  - ✓ The **hierarchy test** verifies that the functional dependencies represented by hierarchies in the multidimensional schema are actually valid on source data
    - If a hierarchy includes a functional dependency that is contradicted by source data, then either (1) a modeling error has been done and the functional dependency should be removed from the multidimensional schema, or (2) source data are faulty, which should be taken care of by ETL
    - Even when no such errors are detected, this test can lead designers to discover denormalization issues they were not aware of in source data, which has a significant impact on ETL design



Functional test of DFM schemas

- It aims at verifying that the multidimensional schemas produced for the data mart effectively support user requirements
  - ✓ The **conformity test** is aimed at assessing how well conformed hierarchies have been designed by evaluating the sparseness of the bus matrix
    - If the bus matrix is very sparse, the designer probably failed to recognize the semantic and structural similarities between apparently different hierarchies (creating a conformed dimension to be shared by multiple facts would be preferable maybe)
    - If the bus matrix is very dense, the designer probably failed to recognize the semantic and structural similarities between apparently different facts (it could be worth merging these facts into a single fact including the union of their measures)

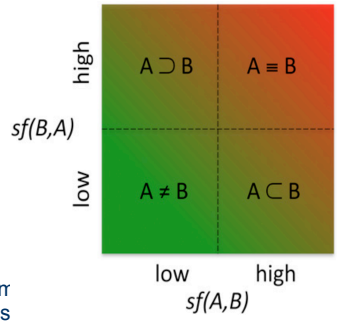
Dimension	TICKET RET.	ORDER	STORE MOVEM.
accounting type		✓	
admin. block reason		✓	
box type		✓	
constraint reason		✓	
counter	✓		
currency	✓	✓	✓
currency type	✓	✓	
discount range	✓		
discount reason	✓		
entity	✓	✓	✓
environment		✓	
exchange	✓	✓	
flag completed order		✓	
flag item block reason		✓	



Functional test of DFM schemas

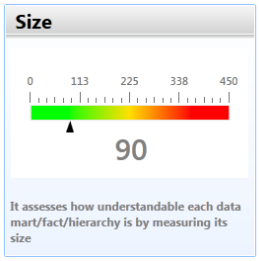
- It aims at verifying that the multidimensional schemas produced for the data mart effectively support user requirements
  - ✓ The **similarity factor** aids designers in evaluating whether the facts in a single data mart have been designed minimally or redundantly, and is computed for each ordered pair of facts
    - Depending on the values of the similarity factors for two facts, the designer can infer that these facts are mainly overlapping, or that one of them is mostly included in the other by aggregation
    - The similarity factor is particularly useful when assessing an existing data warehouse to quickly identify possibly redundant facts
    - When conceptual design is query-driven, several quite similar facts may be erroneously designed, which is easily captured by the similarity factor

	RETAIL TICKETS	ORDERS	STORE MOVEMENTS
RETAIL TICKETS	-	0.65	0.29
ORDERS	0.42	-	0.19
STORE MOVEMENTS	0.71	0.71	-

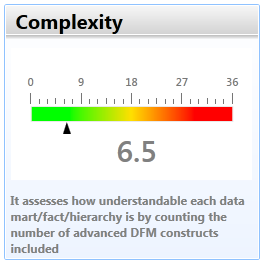


Usability test of DFM schemas

- The **size** metrics counts the total number of attributes in each hierarchy/fact/data mart

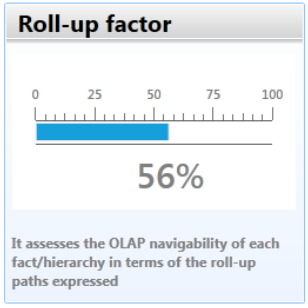


- The **complexity** metrics counts the number of advanced DFM constructs (e.g., multiple arcs, recursive hierarchies, cross-dimensional attributes)
  - ✓ Higher values suggest lower understandability
  - ✓ Hierarchies with size above 25 and complexity above 3 may be difficult to understand for users, while hierarchies with size above 50 and complexity above 3 will most probably create problems with usability and maintainability



Usability test of DFM schemas

- The **roll-up factor** is specifically oriented to hierarchies and aims at evaluating their OLAP navigability in terms of the roll-up paths they express
  - ✓ The roll-up factor of a hierarchy depends on its width and depth: a high value denotes a "deep" hierarchy, while a low value denotes a wide hierarchy
  - ✓ The roll-up factor makes the designer aware that some hierarchies carry low roll-up expressiveness (i.e., they are characterized by short roll-up paths), so that he can check with the user if some roll-up relationships have been forgotten during requirement analysis and conceptual design
  - ✓ Missing existing roll-up relationships is one of the most common mistakes in multidimensional modeling, and leads to a proliferation of degenerate dimensions



Test of the front-end

- Functional test
  - ✓ it should involve a large number of end-users, who are so familiar with application domains that they can detect even the slightest abnormality in data
  - ✓ *balancing test* on a sample basis
- User profile test
- Inference test
  - ✓ evaluate the *disclosure risk*
- Usability test
  - ✓ efficiency (*once users have learned to use the interface, how quickly can they perform tasks?*)
  - ✓ memorability (*when users return to the interface after a period of not using it, how easily can they reestablish efficiency?*)
  - ✓ satisfaction(*how pleasant is it to use the interface?*)
- Performance and stress test

Summary & Conclusion

Summary

- The use of prescriptive design methodologies is very common in DW projects
- In some situations, adopting a more agile approach is beneficial
- Modeling multidimensional data at a conceptual level is a key to streamlining the design process and making it less error-prone
- CASE tools can give a huge contribution in this direction
- An accurate, extensive, and metrics-based testing ensures a top quality for the resulting DW

Practical evidences

A project in the area of pay-tvs with 4WD

- 2 Data marts
  - ✓ *Administration*: 9 facts, 5 releases
  - ✓ *Management* control: 3 facts, 2 releases
- 10 to 26 days for each release
- 6 months overall duration

Benefit	Strategy
Project speed-up	• User involvement (also via web portal) • Extensive prototyping
Reduction of the implementation effort	• Reusing of existing reports and dimension tables
Concise but exhaustive documentation	• DFM as conceptual model
Logical design automation	• Indyco CASE tool

Conclusions

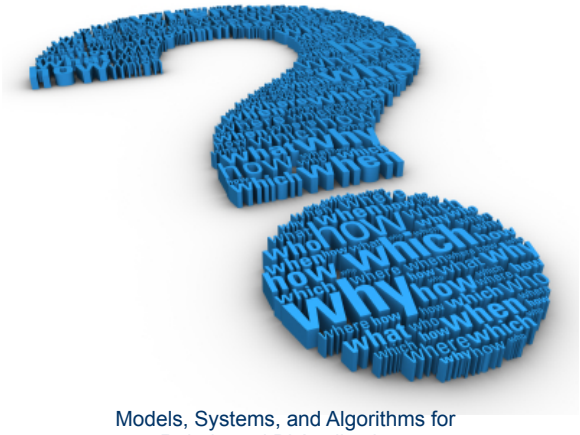
- Agility + conceptual modeling + CASE + testing brings advantages to...
  - ✓ business analysts & data scientists (closer match between requirements and solutions, reduced costs for training, better understanding/exploration of data, self-service BI)
  - ✓ designers (streamlining of design activities, early validation of requirements, effective testing, reduction of the overall development cost)
  - ✓ BI architects (comprehensive documentation, better control over evolution)

References

- Agile Manifesto: Manifesto for agile software development (2010), http://agilemanifesto.org/
- Beyer, M., Richardson, J.: Agile techniques augment but do not replace business intelligence and data warehouse best practice. Tech. Rep. G00201031, Gartner Research (2010)
- M. Golfarelli, S. Rizzi. Data Warehouse Testing: A Prototype-Based Methodology. *Information and Software Technology*, 53(11):1183-1198, 2011
- M. Golfarelli, S. Rizzi, E. Turricchia. Modern Software Engineering Methodologies Meet Data Warehouse Design: 4WD. *Proceedings 13th International Conference on Data Warehousing and Knowledge Discovery*, Toulouse, France, pp. 66-79, 2011
- M. Golfarelli, S. Rizzi. *Data Warehouse Design: Modern Principles and Methodologies*. McGraw-Hill, 2009
- Mazon, J.N., Trujillo, J.: An MDA approach for the development of data warehouses. In: Proc. JISBD, pp. 208–208 (2009)
- Hughes, R.: Agile Data Warehousing: Delivering world-class business intelligence systems using Scrum and XP. IUniverse (2008)
- Sen, A., Sinha, A.P.: A comparison of data warehousing methodologies. Commun. ACM 48(3), 79–84 (2005)
- www.indyco.com

Thank you for your attention

Questions?



Conclusions

- The DFM is used in several projects in public and private companies (e.g., FIAT, Italian Ministry of Justice, University of Bologna, Yoox Group, GEOX, Health Authorities, etc.)
- Today, in big data contexts where companies store large quantities of unstructured data in *data lakes*, conceptual modeling has a key role in ensuring that this huge wealth of information can be effectively explored and analyzed