TATA CONSULTANCY SERVICES

TATA

A model driven approach to enterprise data integration

TATA RESEARCH DEVELOPMENT AND DESIGN CENTRE

Experience certainty.

IT Services
Business Solutions
Outsourcing

15 December 2010 Confidential

Enterprise data management challenge

- A large enterprise has 1000's of databases
  - Not sure what kind of data exists where, and what kind of dependencies exist between them
- Different data stores are produced in different process contexts
  - Not sure what their semantics are
  - Not sure what their quality attributes are
- Lack of a unified view and poor quality severely dent confidence in data assets
  - Affecting operational efficiency
  - Affecting quality of decision making
- Ad hoc, point solutions only exacerbate the problem further
  - They add more pieces to the puzzle

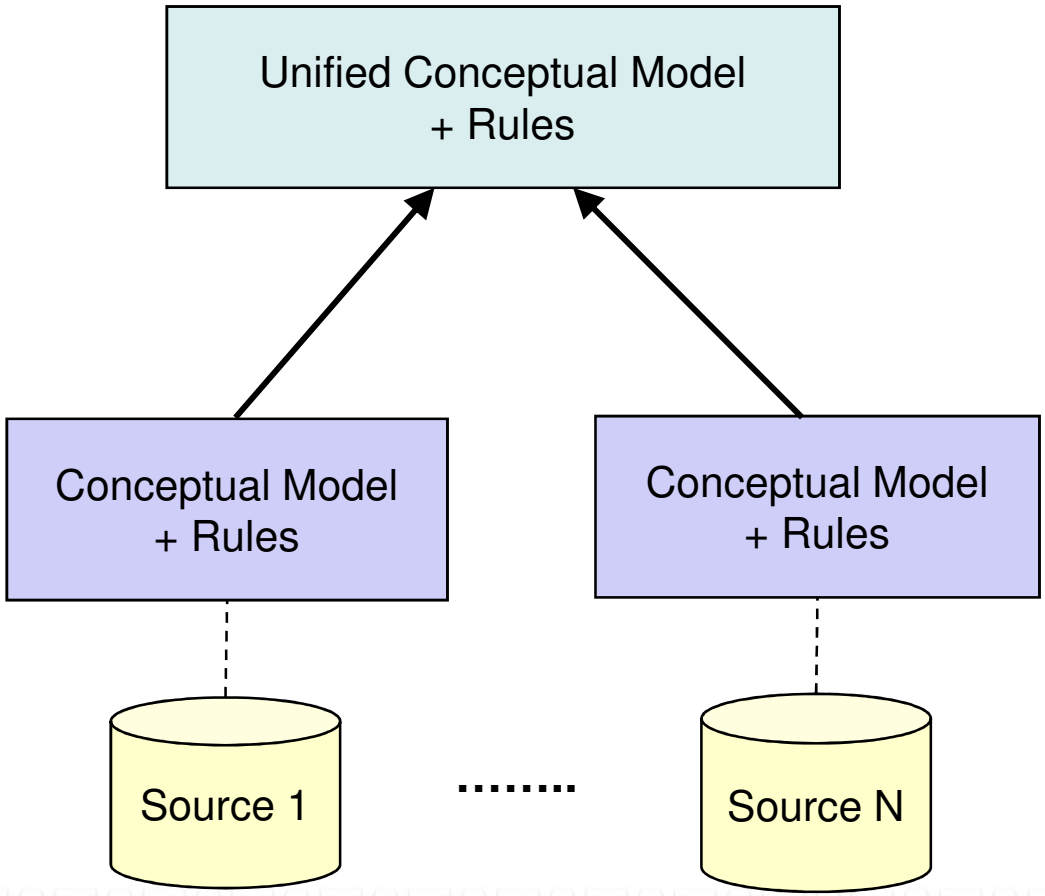
Need a more systematic approach

Model driven approach

A model driven approach to enterprise data integration

- How do we model enterprise data so that we can achieve a unified view of data at the enterprise level?
- Different data sources are produced in different semantic contexts
  - Process, geographical, temporal,…
  - E.g.
    - Customer – current, past, potential?
    - Profit – before tax or after?
    - C1: Employees can work only on projects of their own department
    - C2: Employees can be deputed to work on projects of other departments
- It is critical to model these contexts
  - to interpret and integrate the data correctly
  - to assess its quality

…A model driven approach to enterprise data integration



Global knowledge

Mapping

Context specific knowledge

Raw data (Objects, properties, events,..)

Core infrastructure

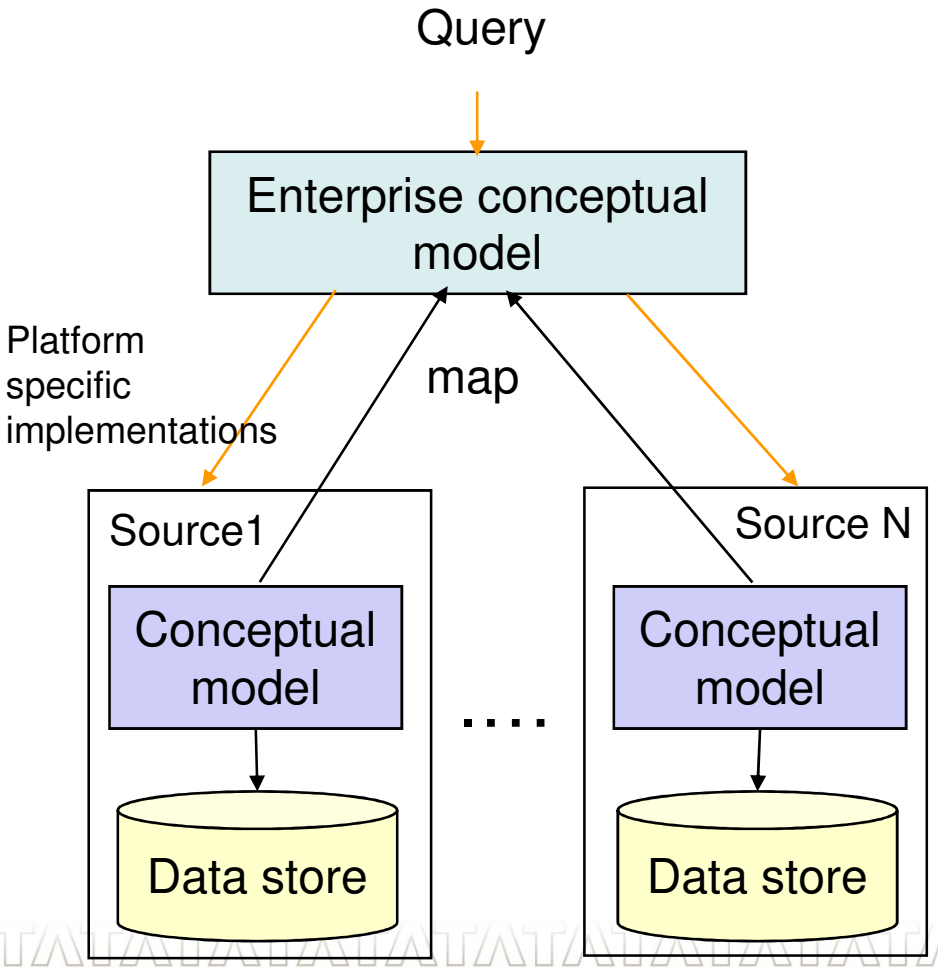
- Models
  - Conceptual models
    - EER or a restricted version of OWL
    - Rule language
      - Similar to SWRL, but extended with aggregation operators
  - Physical models
    - Relational
- Mappings
  - Simple attribute-to-attribute mappings
  - Views – GAV, LAV, GLAV
  - User-defined functions
  - Informal correspondences (with textual description of mappings)

…Core infrastructure

- Core reasoning tools
  - Query translation
    - Given a query on one model translate it into a semantically equivalent one on another mapped model
  - Mapping composition
    - Given Model1<-m1->Model2, Model2<-m2->Model3, compose m1, m2 to produce a mapping between Model1 and Model3.
- Component architecture
  - Modeling, mapping, query translation, query-to-DFG, DFG-to-query, etc
  - Purpose specific data management tools are composed from these components
    - E.g. integration tool, data migration tool, impact analysis tool, etc

Some tools built using modeling approach

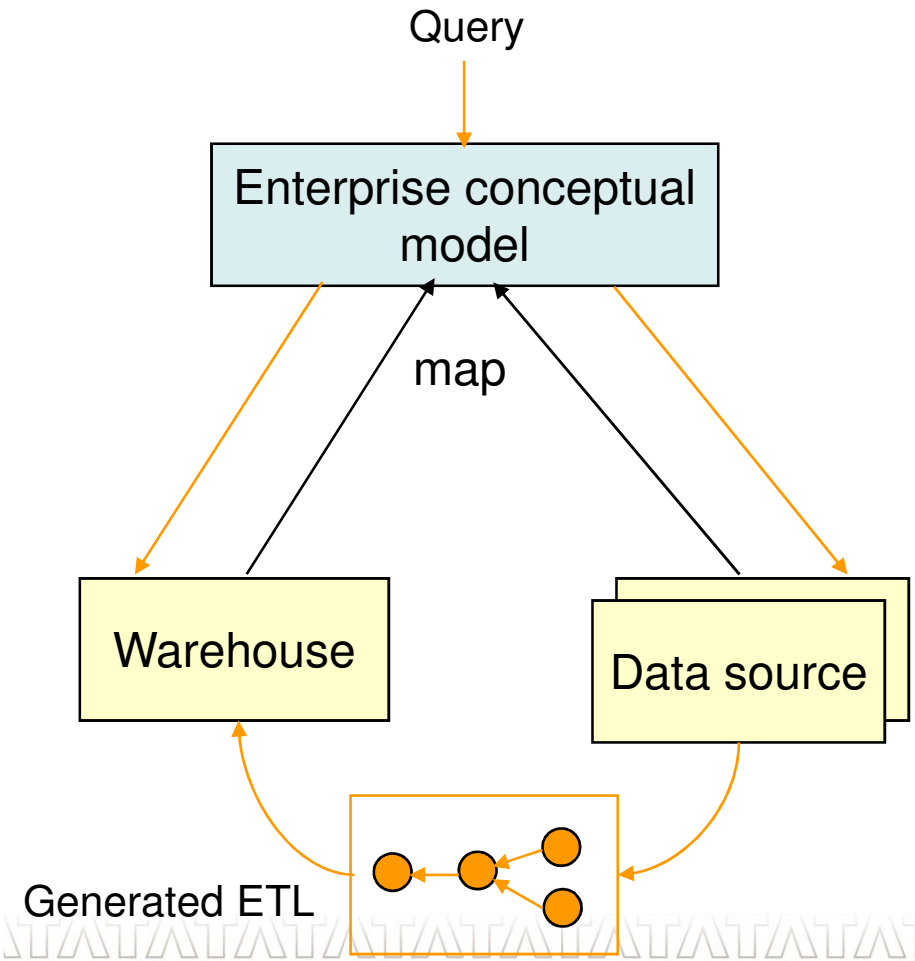
Data integration – a unified view of data



- Provide query interface in terms of enterprise conceptual model
- Generate platform specific implementations
  - DB specific queries
  - ETL specifications
    - Own engine
    - Informatica ETL bridge, ..

Components: modeling, mapping, query translation, query-to-DFG

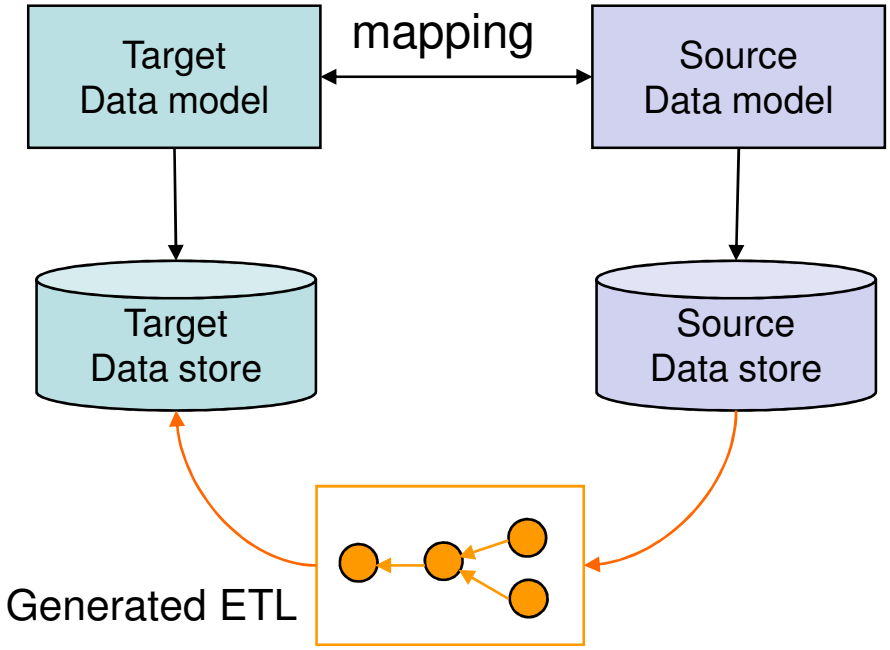
Data integration – hybrid architecture



- Specify warehouse schema as a view over the unified enterprise conceptual model
- Generate ETL to populate warehouse from sources
- Provide a unified query interface in terms of the enterprise conceptual model
  - Go to warehouse for data that exists there and to sources for other data
    - Pick optimal set of aggregate views
  - As requirements change, this mix can change transparently

Components: modeling, mapping, mapping composition, query translation, query-to-DFG

Data migration



- Map source data model to target data model
- Generate ETL to migrate data from source to target
- Generate migration programs for data access code migration
  - Query migration

Components: modeling, mapping, query translation, query-to-DFG

Open Issues

Complexity vs. expressive power

- Query complexity vs. data complexity
  - We wanted to keep data complexity polynomial
  - SQL reducibility is a goal
- But SQL reducible subsets are not expressive enough
  - E.g. OWL Lite
- Partial solutions vs. complete solutions
  - Completeness sacrificed in favor of expressiveness
  - But safety guaranteed
  - E.g. comparison predicates, aggregation, functional dependencies
    - Data complexity is known to be NP
    - We use SQL reduction, so obviously there is no completeness guarantee

Can we state conditions on the query and models that tell us when a given rewriting is guaranteed to be complete?

Static vs. dynamic context

- We have just one fixed context model per data source
  - But context can vary from instance to instance even within the same entity
  - E.g. model: Company, Department, Employee, Project
    - Company X: Employees can work only on projects of their own department
    - Company Y: Employees can be deputed to work on projects of other departments
- To implement this, we need two levels of rules
  - Rules to select applicable context for a given instance
  - Application of context specific rules

How do we do this efficiently is an open issue.

Integrating quality attribute processing

- How do we deal with quality issues such as uncertainty, etc?
- Plenty of research on probabilistic databases and uncertainty management
  - E.g. Stanford TRIO - uncertainty and provenance management on top of relational model
  - They assume confidence values are already recorded in the database
  - But assessing uncertainty of 'base facts' itself is a complex reasoning task requiring context knowledge
  - Can this be integrated into the rest of the query processing framework?
- Can uncertainty management be used to resolve conflicts better?
  - In integration, quite often we end with conflicting values
  - Uncertain which value to choose, but forced to choose one
  - Can uncertainty management techniques provide a better solution?

Thank you!