

Meta Data Management

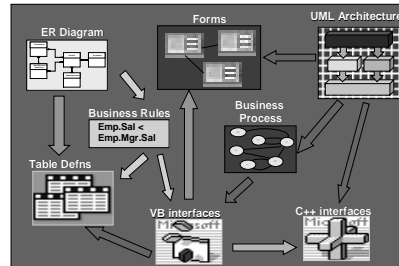
Philip A. Bernstein
Sergey Melnik
{philbe, melnik}@microsoft.com

Microsoft Research

Seminar presented at ICDE, Boston, April 1, 2004
© 2004 Microsoft Corporation. All rights reserved.

Meta Data Management

- Meta data = structural information
 - > DB schema, interface defn, web site map, form defs, ...



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

2

Meta Data Problems

- Many data management applications primarily involve transformations of structured data
- Data translation
- Schema evolution
- XML message translation
- Application integration
- Data warehouse loading
- ER/UML design tools
- Wrapper generation for SQL
- UI / 4GL generation
- Dependency tracking
- Lineage tracing
- Info resource mgmt
- Binding, renaming
- Software build (make)
- Configuration mgmt

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

3

Outline

- Introduction
- Meta data problems
- Design patterns
- Solution templates
- Research background
- Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

4

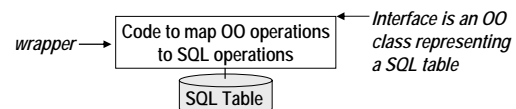
Why Meta Data is Important

- Many DB problems are easier to solve by manipulating meta data
 - > Instead of writing code
 - > Instead of manipulating data directly
- Meta-data-based solutions all involve models (schemas) and mappings
 - > Mappings - data transformations, queries, dependencies, ...
 - > Model, manipulate, and generate them
 - > Usually, generate code from them

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

5

Example: Object-Oriented Wrapper for SQL Tables

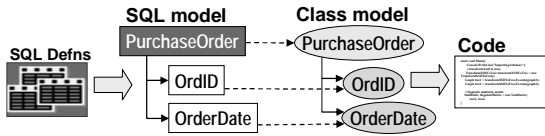


- Manually program a wrapper for each table
- This is very repetitive work
- So you write a program to generate a wrapper for each table

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

6

OO wrapper for SQL (cont'd)



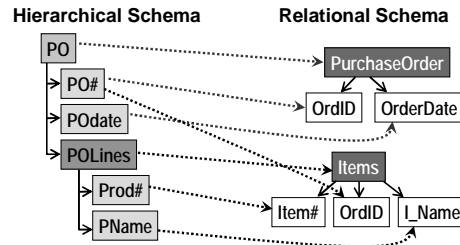
- The wrapper generator does the following:
 - Imports each table definition into a model
 - Generates a model for the class wrapper
 - Generates a mapping from table to class
 - Generates code from the class model and mapping.

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

7

Example – Data Translation

- Translate data from one data model to another
- Either write a program or generate it



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

8

Meta-data-Speak

Meta-data-Speak	English Example
meta-meta-model = meta- meta-meta data	Built-in types (usually hard-coded)
metamodel = meta-meta data	Schema for "Table," "Column", "Key," ...
model = meta data	Schema for the Employee Table
data	Employee Table

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

9

Outline

- Introduction
- ➔ Meta data problems
- Design patterns
- Solution templates
- Research background
- Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

10

Meta Data Solution Template

- 1 Get a data manager for models and mappings

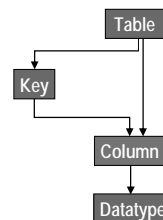
- Usually, it's an object manager
 - OO programming language
 - OODB
- Hence, meta-metamodel is the object manager's built-in types
 - Classes, attributes, methods, objects
 - Plus operators to manipulate them, such as NewClass, NewAttribute, NewObject, WriteAttribute

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

11

Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)



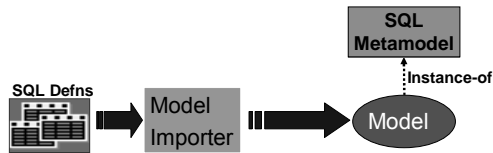
If the meta-metamodel is OO, then the metamodel consists of class definitions

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

12

Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)
- 3 Build a model importer for each metamodel

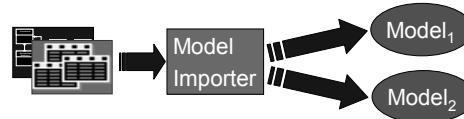


P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

13

Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)
- 3 Build a model importer for each metamodel
- 4 Invoke model importer(s)



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

14

Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)
- 3 Build a model importer for each metamodel
- 4 Invoke model importer(s)

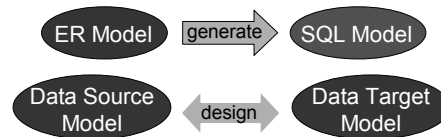
Problem	Model ₁	Model ₂
Data translation	source schema	target schema
Msg translation	source format	target format
App integration	source interfaces	target interfaces
DW loading	source schema	DW schema

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

15

Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)
- 3 Build a model importer for each metamodel
- 4 Invoke model importer(s)
- 5 Generate or design mappings



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

16

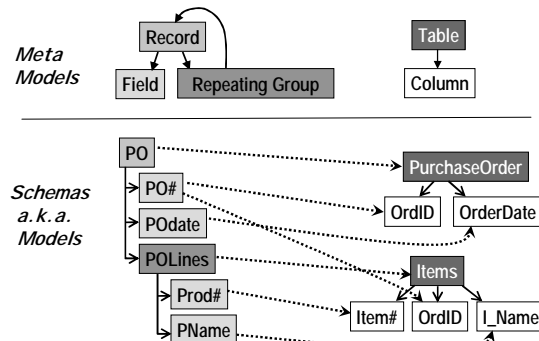
Meta Data Solution Template

- 1 Get a data manager for models and mappings
- 2 Design metamodel(s) (e.g., for SQL schemas)
- 3 Build a model importer for each metamodel
- 4 Invoke model importer(s)
- 5 Generate or design mappings
- 6 Generate code: data / msg transl'n script, app wrapper, ETL script, view defn's, etc.

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

17

Example – Data Translation



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

18

Generated data translation script

```

For each [po#, poD, poL] in PO
  Insert [po#, poD] into PurchaseOrder
  For each [prod#, pN] in poL
    Insert [prod#, po#, pN] into Items
  End
End
  
```

Schemas a.k.a. Models

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 19

Meta Data Problems

- Data translation
- OO or XML wrapper generation for SQL DB
- User-Interface / 4GL-program generation
- Design tool support (DB, UML, ...)
 - Model generation, reverse engineering
 - Round-trip engineering
- Schema evolution (applies to all scenarios)
- XML message translation for e-commerce
- Integrate custom apps with commercial apps

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 20

Meta Data Problems (cont'd)

- Data warehouse loading (clean & transform)
- Lineage tracing (provenance)
- Information resource management
- Dependency tracking
 - Impact analysis
 - Navigation between tools
- Binding, renaming
- Software build (make)
- Version and configuration management
 - Release management
 - Product data management

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 21

Meta Data Solutions

- They strongly resemble one another
- We characterize that resemblance
 - Prototypical problems, or *design patterns*
 - Solution specifications, or *solution templates*
 - Primitive solution steps, or *operators*
- Goals
 - A methodology to solve meta data problems
 - Ultimately, operator implementations to turn solution templates into solution programs

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 22

Outline

- Introduction
- Meta data problems
- ➔ Design patterns
- Solution templates
- Research background
- Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 23

Meta Data Design Patterns

- Design pattern – a problem description consisting of
 - Input models and mappings
 - Output models and mappings
 - Criteria for the output to be correct
 - An application specializes it to meta models and mapping languages
- Solution template – a sequence of operators producing the desired output
- Operators – a single step that computes a model and/or mappings

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 24

Operators

- $map = Match(M_1, M_2)$
 - Return a mapping between the two models
- $\langle M_2, map_{12} \rangle = ModelGen(M_1, metamodel_2)$
 - Return a model M_2 that is expressed in $metamodel_2$ and is equivalent to model M_1
- $\langle M_3, map_{13}, map_{23} \rangle = Merge(M_1, M_2, map)$
 - Return the union of models M_1 and M_2
- $map_3 = Compose(map_1, map_2) = map_1 \circ map_2$
 - Return the composition of map_1 and map_2 , which is a mapping from map_1 's domain to map_2 's range.

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

25

Operators (cont'd)

- $map_3 = Confluence(map_1, map_2) = map_1 \oplus map_2$
 - Return the "merge" of mappings map_1 and map_2
- $\langle M_2, map_{12} \rangle = Extract(M_1, map)$
 - Return the sub-model of M_1 that participates in the mapping map
- $\langle M_2, map_{12} \rangle = Diff(M_1, map)$
 - Return the sub-model of M_1 that does not participate in the mapping map

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

26

Design Patterns

- Meta Modeling
 - Model Mapping
 - Model Generation
 - Model Integration
 - Mapping Composition
 - Mapping Alignment
 - Change Propagation
 - Model Reintegration
- } Single Operator Solutions

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

27

Meta Modeling

- Design pattern – develop a representation (i.e. metamodel) for models and mappings
- Applications – they all depend on this
- Solution template
 - Design a metamodel
 - Write Import & Export functions
 - ImportSQL, ImportXSD, ImportERD, ...
 - Today, it is manual engineering design
 - Design once and reuse often

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

28

Meta Modeling (cont'd)

- The Import function for models
 - Parse text
 - Copy elements of the parsed form into a model that conforms to its metamodel
- The Import function for mappings
 - Same as models but may require more semantic analysis
 - E.g., program dependencies, data lineage
 - For some languages and mapping metamodels, Export is hard (e.g., XSLT)

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

29

Model Mapping

- Design pattern – Design a mapping between two models and generate code from it

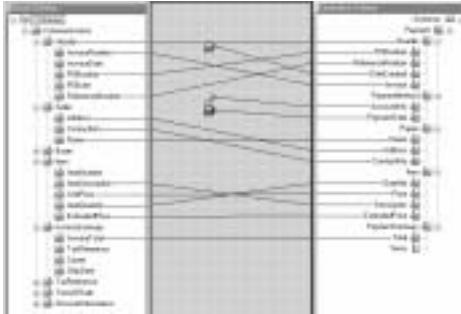


- Applications
 - Data translation
 - XML message translation for e-commerce
 - Integrate custom and commercial apps
 - Data warehouse extract, transform & load
- Solution templates
 - $map = Match(M_1, M_2)$; $Export(map)$
 - Mapping reuse: $Compose$, $Confluence$

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

30

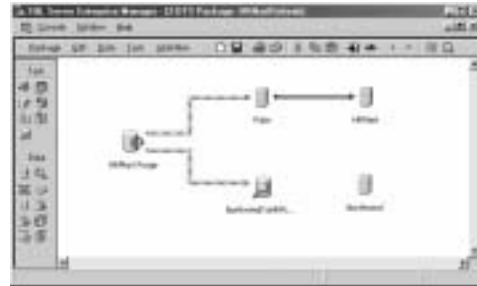
An XML Mapping Tool



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

31

A Data Warehouse Loading Tool



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

32

Model Generation

- Design pattern – Given a model, generate an equivalent model in another metamodel



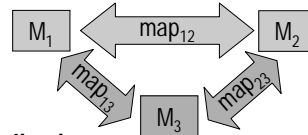
- Applications
 - Wrapper generation (SQL → OO or XML)
 - Design tools (ER → SQL, SQL → ER)
 - UI/4GL generation
- Solution template
 - $\langle M_2, map \rangle = \text{ModelGen}(M_1, \text{metamodel}_2); \text{Export}(M_2)$
 - ModelGen often needs human guidance

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

33

Model Integration

- Design pattern – Given two models, develop a model that subsumes both of them



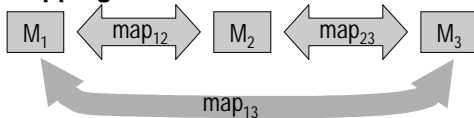
- Applications
 - View integration
 - Data integration
- Solution template
 - $\langle M_3, map_{13}, map_{23} \rangle = \text{Merge}(M_1, M_2, map); \text{Export}(M_3, map_{13}, map_{23})$

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

34

Mapping Composition

- Design pattern – Compose two given mappings



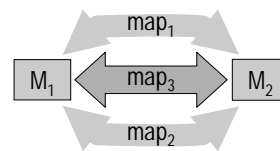
- Applications
 - Processing queries on views
- Solution templates
 - $map_{13} = \text{Compose}(map_{12}, map_{23})$
 - Answering queries using views (LaV), Query modification (GaV), GLaV

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

35

Mapping Alignment

- Design pattern – Align two mappings between the same pair of models



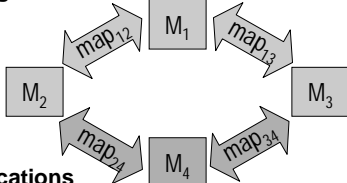
- Applications
 - P2P query processing, mapping design
- Solution template
 - $map_3 = \text{Confluence}(map_1, map_2)$

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

36

Model Reintegration

- Design pattern – Given a model and mappings to two modified versions of the model, produce a merged model



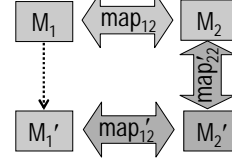
- Applications
 - Parallel development
- Solution template
 - Multistep application of many operators

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

37

Change Propagation

- Design pattern – Given two models and a mapping. One model changes. Fix the mapping and other model.



- Applications
 - Schema evolution, interface evolution, ...
 - Required maintenance for all meta data problems
- Solution template
 - Requires all of the operators

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

38

Outline

- Introduction
- Meta data problems
- Design patterns
- ➔ Solution templates
 - Model reintegration
 - Change propagation
- Research background
- Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

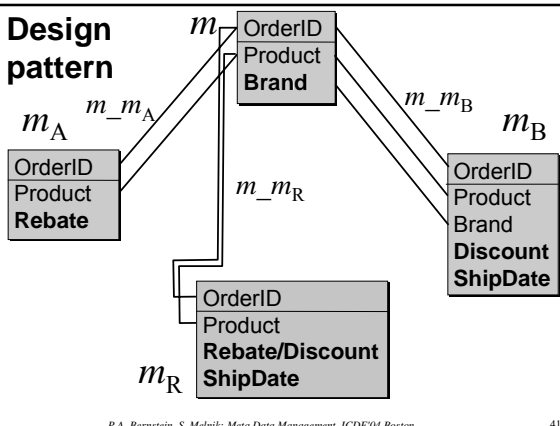
39

Model reintegration

- Design pattern
 - Reconcile independent changes
 - All changes of each model
 - No “duplicate additions”
- Simplified example
 - “Additions” = add model element (also: drop constraints, reorg. model)
 - “Deletions” = delete model element (also: add constraints, reorg. model)
 - Mappings shown as lines betw. elements

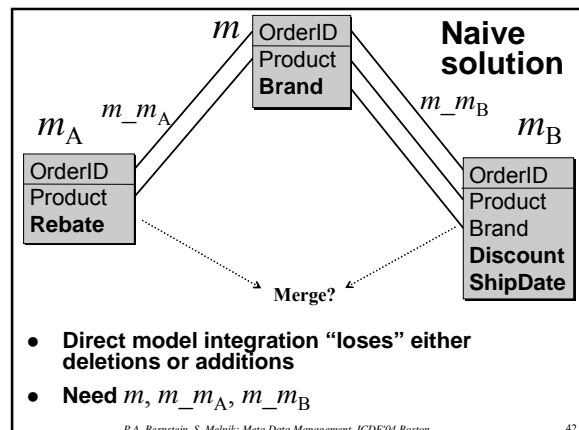
P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

40



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

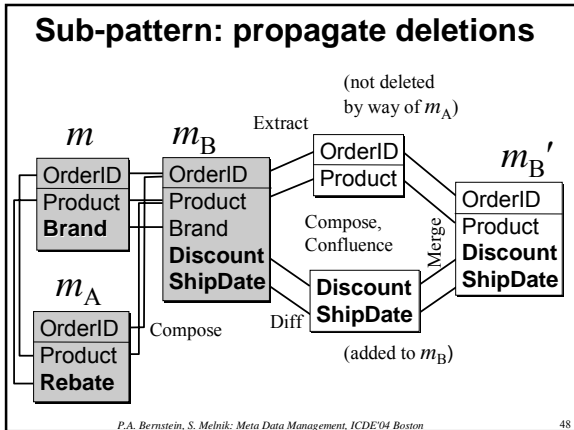
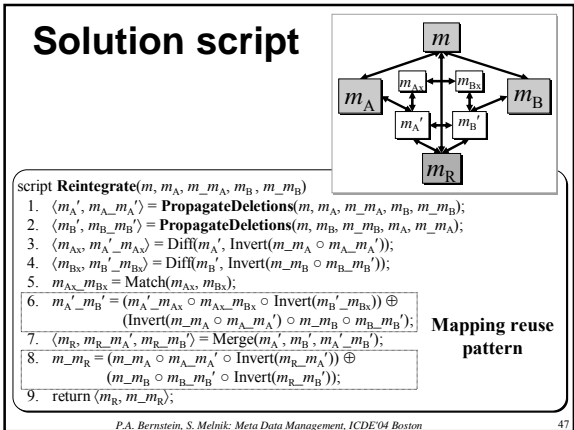
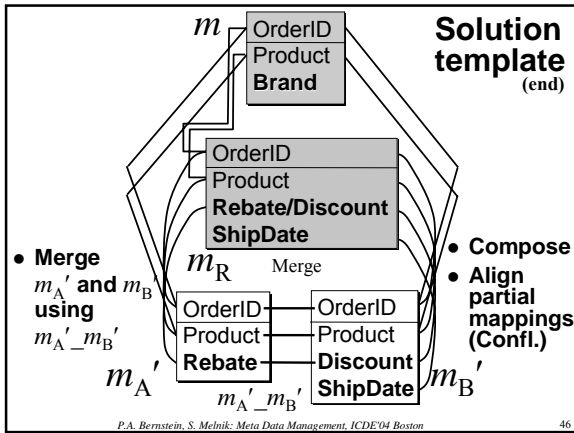
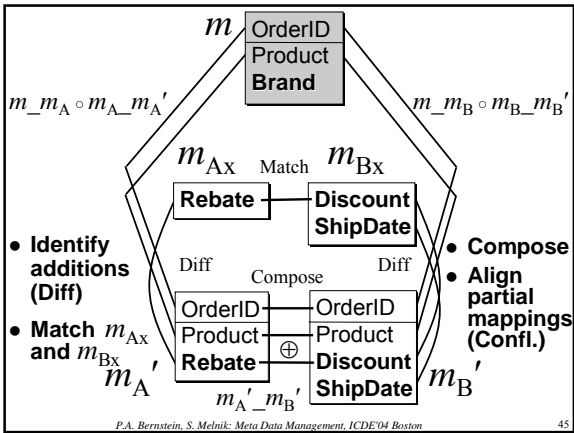
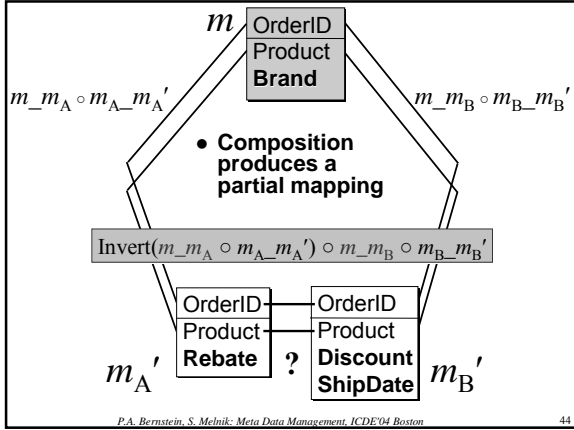
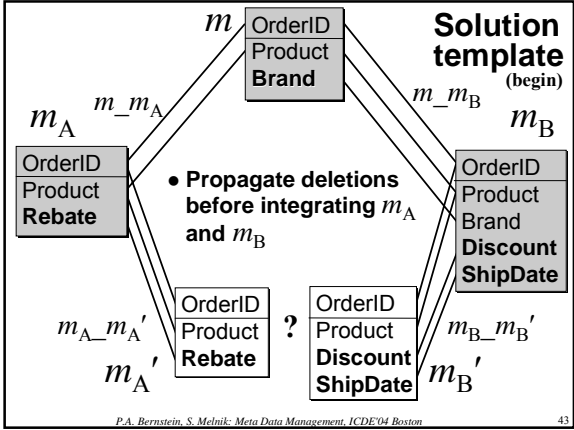
41



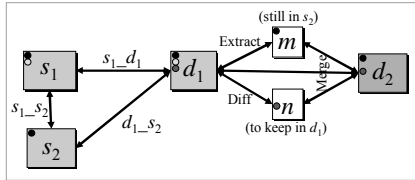
- Direct model integration “loses” either deletions or additions
- Need m, m_{m_A}, m_{m_B}

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

42



Solution script



```

script PropagateDeletions(s1, d1, s1_d1, s2, s1_s2)
1. d1_s2 = Invert(s1_d1) o s1_s2;
2. (m, d1_m) = Extract(d1, d1_s2);
3. (n, d1_n) = Diff(d1, Invert(s1_d1));
4. (d2, d2_m, d2_n) = Merge(m, n, Invert(d1_m) o d1_n);
5. d1_d2 = (d1_m o Invert(d2_m)) o (d1_n o Invert(d2_n));
6. return (d2, d1_d2);
    
```

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 49

Outline

- Introduction
- Meta data problems
- Design patterns
- Solution templates
 - Model reintegration
 - ➔ Change propagation
- Research background
- Wrap up

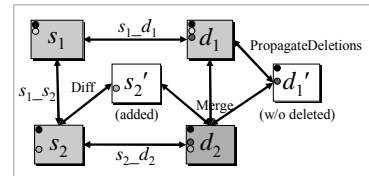
P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 50

Change propagation

- Solution template
 - Propagate deletions
 - Include additions
 - Merge result

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 51

Change propagation

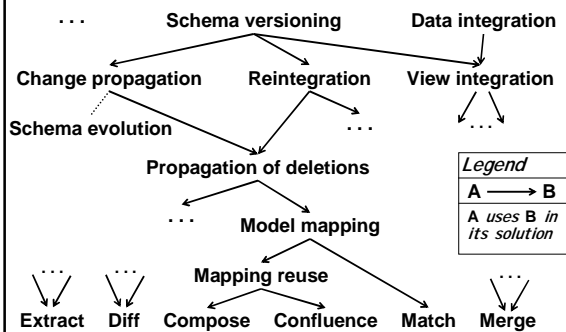


```

script PropagateChanges(s1, d1, s1_d1, s2, s1_s2)
1. (d1', d1_d1') = PropagateDeletions(s1, d1, s1_d1, s2, s1_s2);
2. (s2', s2_s2') = Diff(s2, Invert(s1_s2));
3. (d2, d2_m, d2_d1') = Merge(s2', d1', Invert(s1_s2 o s2_s2') o s1_d1 o d1_d1');
4. d1_d2 = (Invert(s1_d1) o s1_s2 o s2_s2' o Invert(d2_s2')) o (d1_d1' o Invert(d2_d1'));
5. s2_d2 = (Invert(s1_s2) o s1_d1 o d1_d1' o Invert(d2_d1')) o (s2_s2' o Invert(d2_s2'));
6. return (d2, d1_d2, s2_d2);
    
```

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 52

First-cut taxonomy of patterns



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 53

Outline

- Introduction
- Meta data problems
- Design patterns
- Solution templates
 - ➔ Research background
- Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston 54

Foundation of operators

- **View & schema integration (Merge)**

- Batini et al: ACM Comp. Surveys, 18 (4), '86
- Beeri, Milo: ICDT '99
- Biskup, Convent: SIGMOD '86
- Buneman et al: EDBT '92
- Casanova, Vidal: SIGMOD '83
- Lenzerini: PODS '02
- Motro: IEEE TSE, 13:7, '87
- Pottinger, Bernstein: VLDB '03
- Rosenthal, Reiser: TODS 19 (2), '94
- Spaccapietra, Parent: TKDE 6 (2), '94

- **Composition of queries, views, GLAV maps; query processing**

- Abiteboul, Vianu, Hull: Addison-Wesley, 1995
- Fernandez et al: TODS 27 (4), '02
- Madhavan, Halevy: VLDB '03
- Papakonstantinou et al, SIGMOD '99
- Shanmugasundaram et al, VLDB '01
- Shu et al, TODS 2 (2), '77

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

55

- **View selection, answering queries using views (Extract, Confluence)**

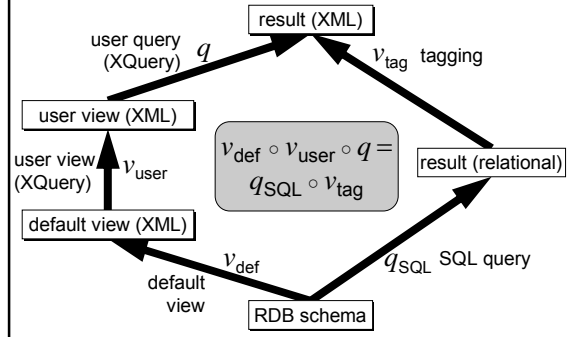
- Agrawal et al: SIGMOD '01
- Chen et al: IDEAS '02
- Chirkova et al: VLDB '01
- Gupta et al: PODS '03
- Halevy: VLDB J. 10:4, '01
- Li et al: ICDT '01
- Theodoratos et al: DKE 39 (3), '01

- **View update, view complement (Diff)**

- Bancilhon, Spyratos: TODS 6:4, '81
- Cosmadakis, Papadimitrou: J. ACM '84
- Dayal, Bernstein: VLDB '78
- De Amo et al: IDEAS '00
- Hegner: J. Comp. Sys. Sci., '94
- Keller, Ullman: SIGMOD '84
- Lechtenbörger, Vossen: TODS 28:2, '03

Example: composition

Shanmugasundaram et al, VLDB '01

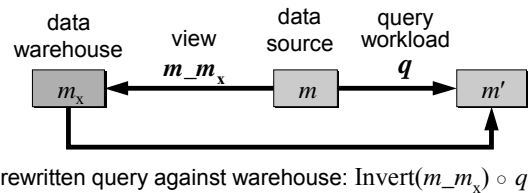


P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

56

Example: Extract

- **View selection, answering queries using views**

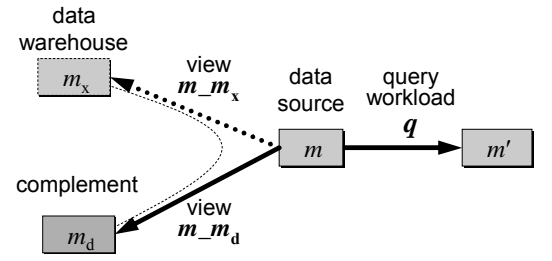


P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

57

Example: Diff

- **View complement: view update, self-maintainability**

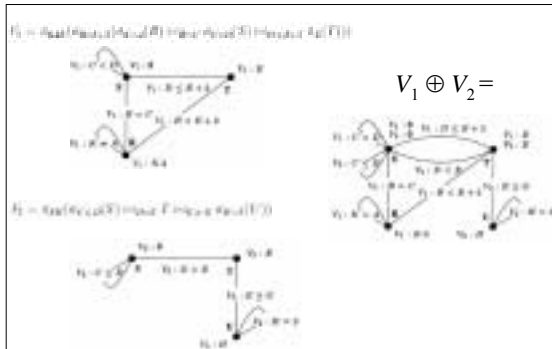


P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

58

Example: Confluence

Theodoratos et al, DKE 39 (3), '01



P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

59

Outline

- Introduction
- Meta data problems
- Design patterns
- Solution templates
- Research background
- ➔ Wrap up

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

60

The Commercial World

- **Books for IT professionals**
 - A. Tanenbaum: Metadata Solutions, Addison-Wesley, 2001
 - D. Marco: Building and Managing the Meta Data Repository, Wiley, 2000
- **Standards-**
 - UML, MOF, CWM (OMG)
 - XML, RDF, XML Schema, OWL (W3C)
- **Products and tools**
 - Modeling: IBM Rational Rose, Visio, CA AII Fusion, Borland Together
 - General meta data managers: CA Advantage, Microsoft Meta Data Services, MetaIntegration
 - Meta data services in data warehousing ETL tools: Informatica, Ascential, ETL, Data Advantage, ...

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

61

The Research World

- **Model Management**
 - A computational meta data framework based on models, mappings, and the operators described here (Match, Merge, Compose, ...)
- **Meta Data is a very active research area**
 - Papers coming from many DB research groups
 - Some are problem-focused (e.g. data integration)
 - Some are operator-focused (e.g. Match, Merge)

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

62

Summary

- **Many DB problems are easier to solve by manipulating meta data**
- **Meta data problems and solutions strongly resemble one another**
- **Methodology: Use design patterns, solution templates, and operators to simplify development of meta data applications**
- **There is much research to be done**

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

63

References

- <http://research.microsoft.com/db/ModelMgt>
- **Overview**
 - Bernstein, CIDR 2003
 - Bernstein, Halevy, Pottinger, SIGMOD Record, Dec. 2000
- **Implementation**
 - Melnik, Rahm, & Bernstein, SIGMOD 2003 & J. Web Semantics 1, 2003
- **Data Warehouse Examples**
 - Bernstein & Rahm, ER 2000
- **Match Operation**
 - Survey: Rahm & Bernstein, VLDB J., Dec. 2001
- **Merge Operation**
 - Pottinger & Bernstein, VLDB 2003

P.A. Bernstein, S. Melnik: Meta Data Management, ICDE'04 Boston

64



© 2004 Microsoft Corporation. All rights reserved.
This presentation is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.