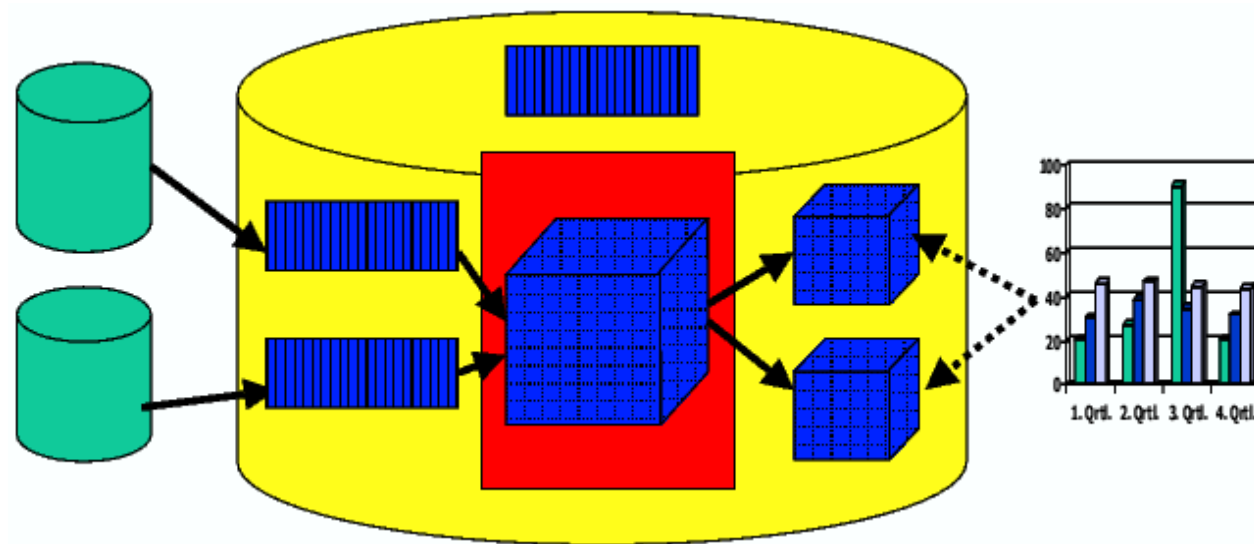
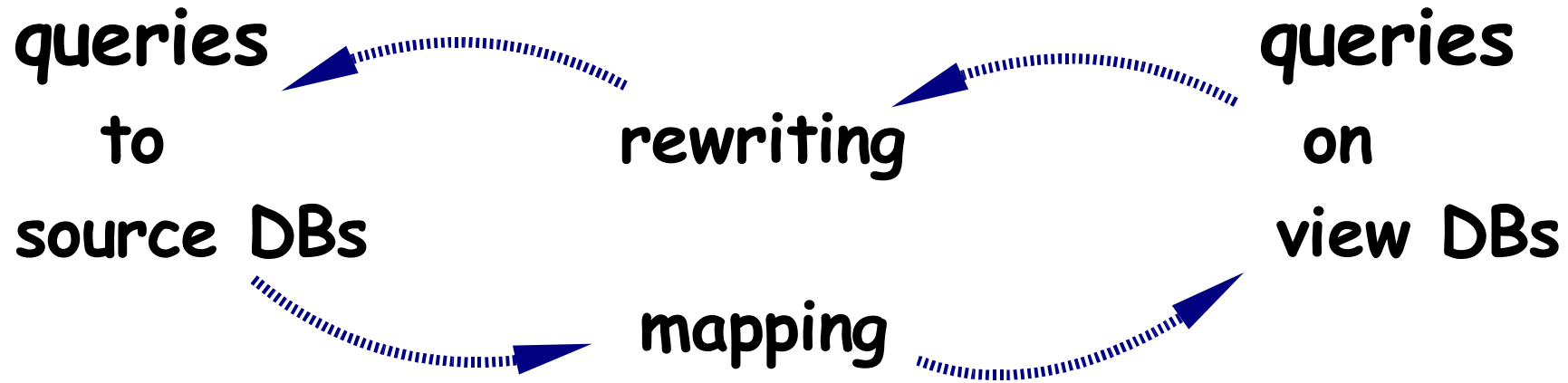
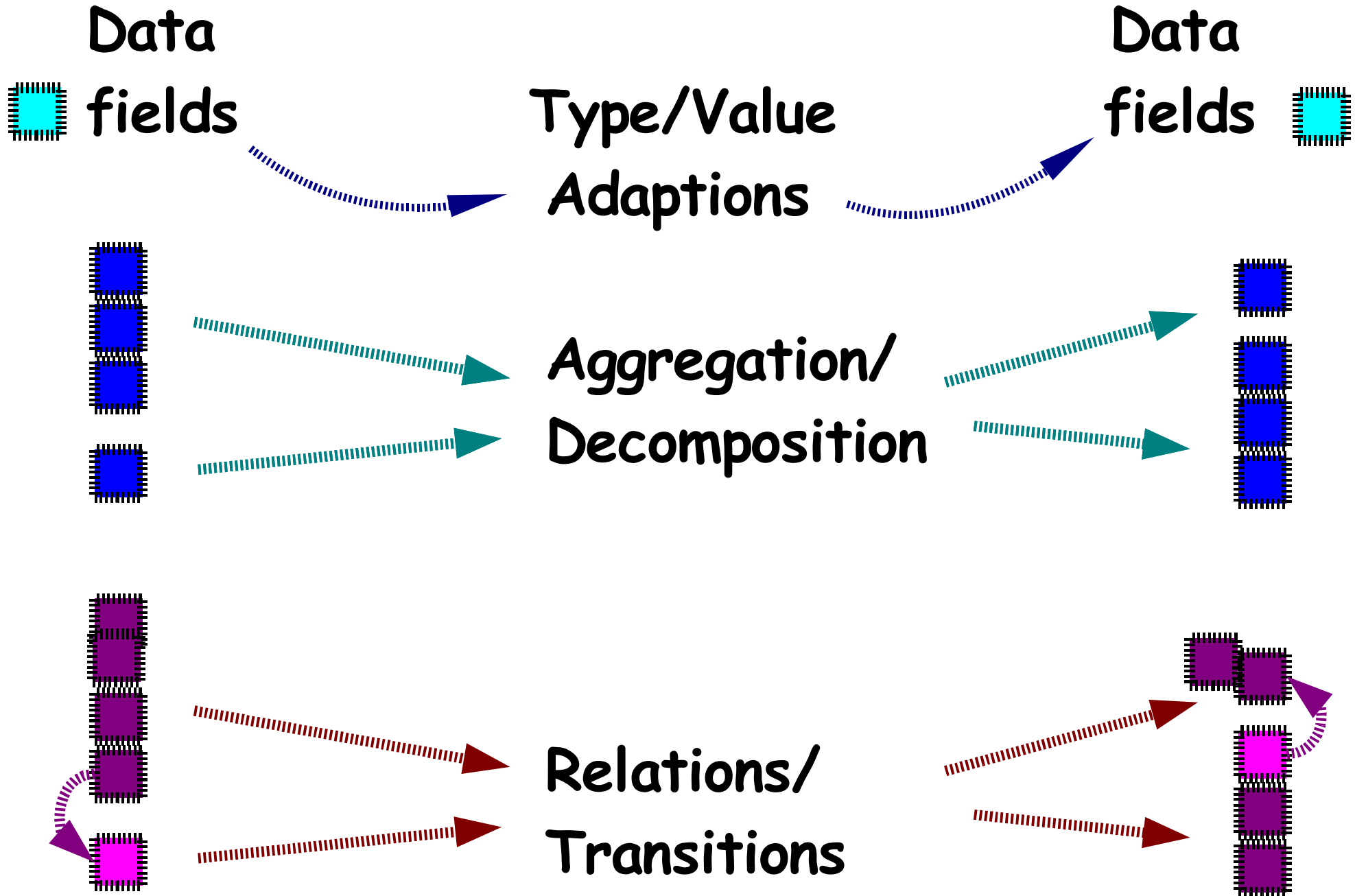


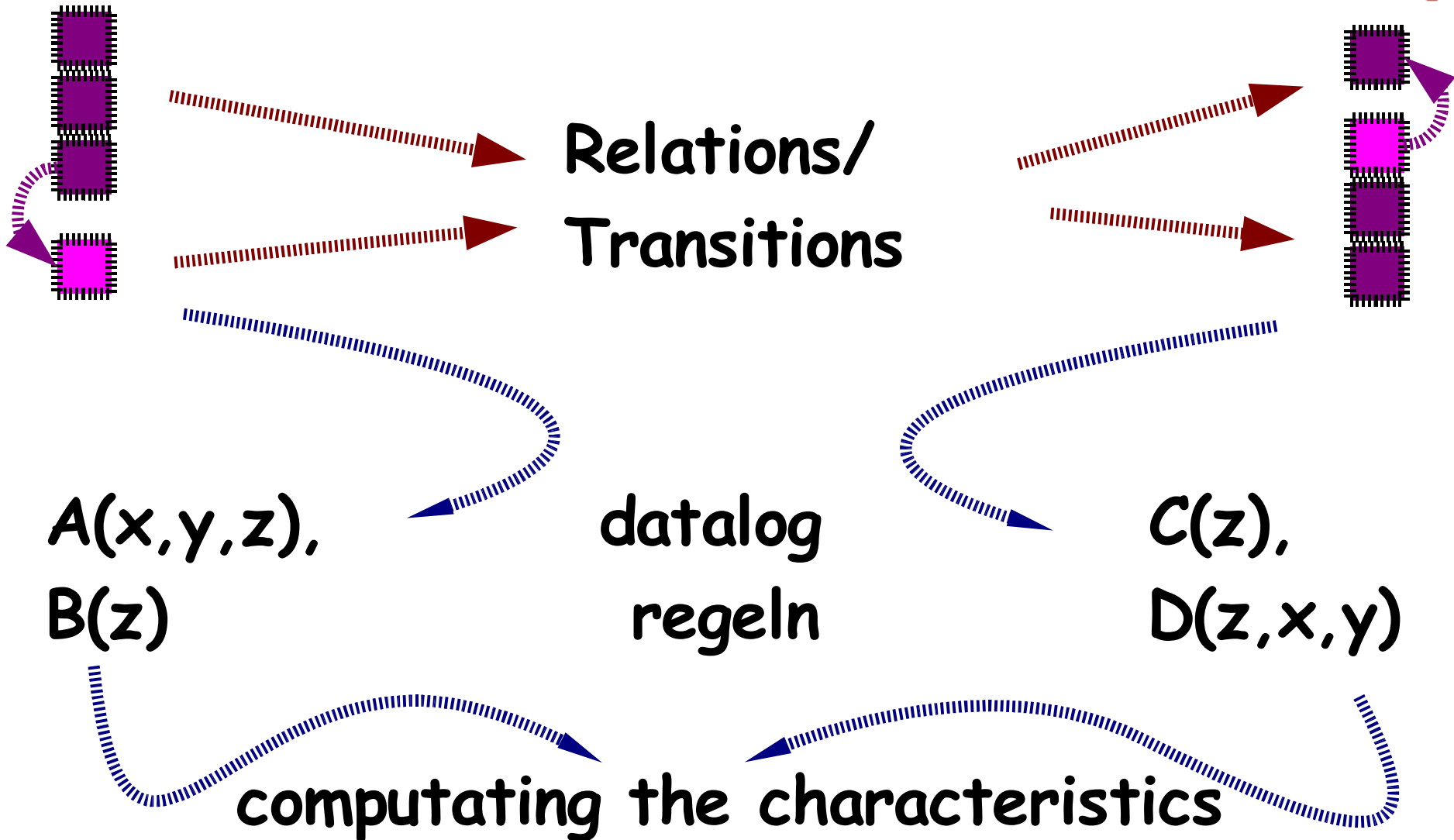
Übersicht Schema Anpassung



Übersicht Schema Anpassung



Übersicht Schema Anpassung



KD: key dependencies, ID: inclusion dependencies, ED: exclusion dependencies
CM: query containment RQ: redundant query optimizations ...

Übersicht Schema Anpassung

$A(x, y, z),$

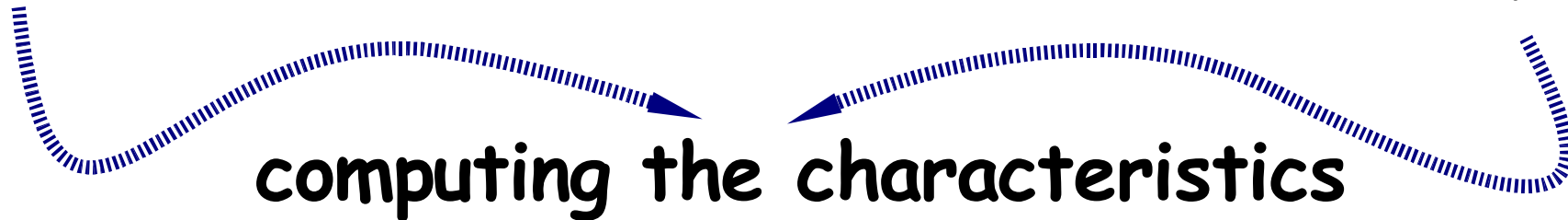
datalog

$C(z),$

$B(z)$

ruleset

$D(z, x, y)$



Relations and Values are

- explicit in RDMBs Tables
- implicit in Transformation Procedure Heads
- temporary in memorized results and some materialized queries.

--> computable

Übersicht Schema Anpassung

$A(x,y,z),$
 $B(z)$

datalog
ruleset

$C(z),$
 $D(z,x,y)$

computed
mapping ruleset



queries on
source DB
wrappers

>> unfolding <<

view
table

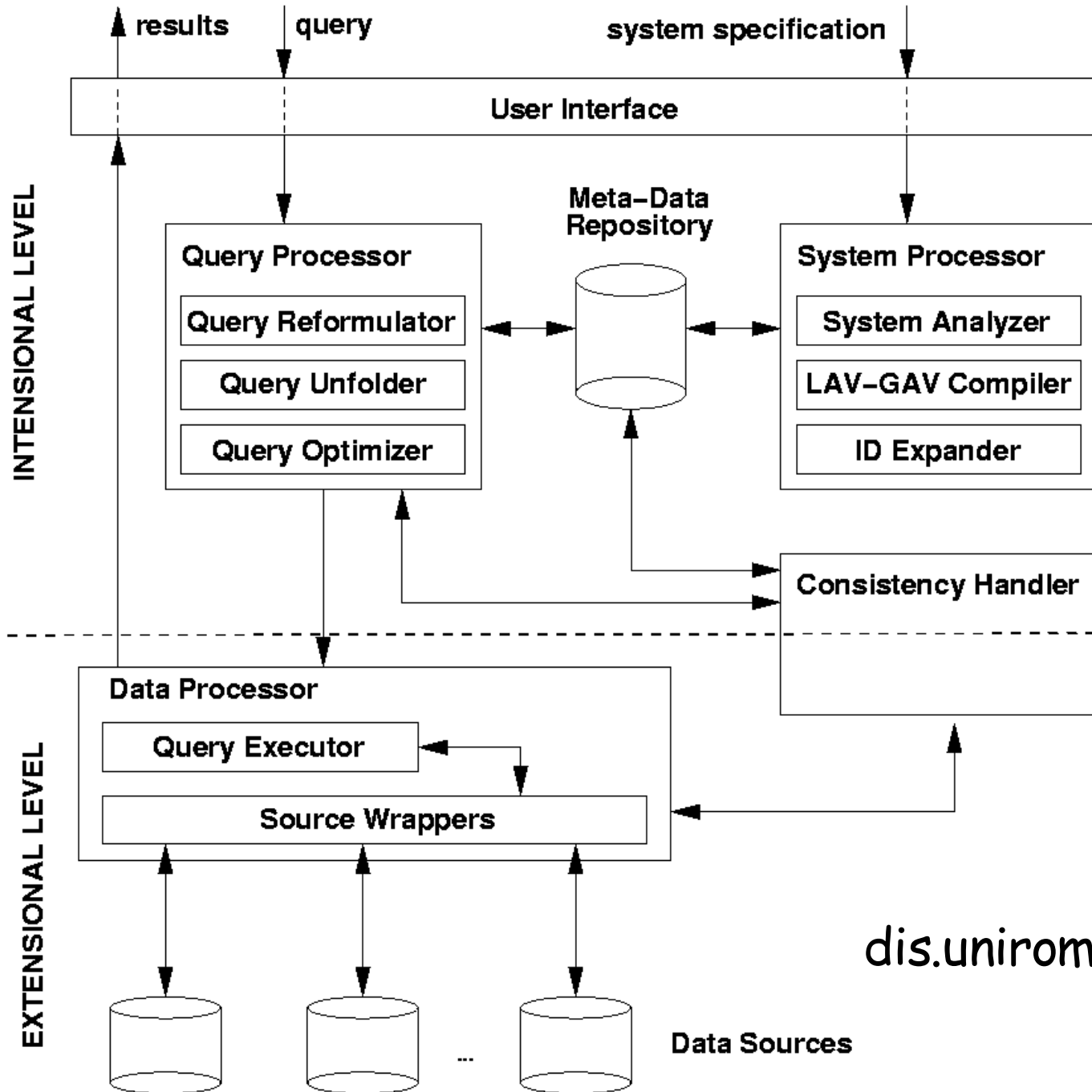
transformation
procedures

cached queries

aggregation



Übersicht Schema Anpassung



dis.uniroma1.it/~disatdis/implementation.html

UPDATES

22-02-2003 INIZIO PROGETTAZIONE E SVILUPPO DELL'INTERFACCIA WEB DEL PROTOTIPO DEL SISTEMA

18-02-2003 COMPLETAMENTO VERSIONE PRELIMINARE DELL'ALGORITMO **ID-REWRITE**

10-02-2003 COMPLETAMENTO ALGORITMO **LAV-GAV COMPILE**

02-02-2003 COMPLETAMENTO ALGORITMO **INVERSE RULES, CHASE RULES E QUERY RECTIFICATION**

25-01-2003 COMPLETAMENTO ALGORITMO **ID-CLOSURE** PER IL CALCOLO DELLA CHIUSURA DEL SET DI VINCOLI INCLUSIONI

18-01-2003 COMPLETAMENTO PACKAGE **DISPROJECT . DISPARSER** - INIZIO DELLA FASE DI TEST

16-01-2003 PUBBLICAZIONE DELLE SEGUENTI CLASSI APPARTENENTI AL PACKAGE **DISPROJECT . DISPARSER**:

- **DISQueryParser**
- **DISQuerySetParser**

DI CUI SI CONSIGLIA VIVAMENTE DI CONSULTARE LA SEGUENTE [PAGINA DI ISTRUZIONI](#)

09-01-2003 NELLA VERSIONE DISPONIBILE PER IL DOWNLOAD SONO PRESENTI I SEGUENTI AGGIORNAMENTI:

- Classe per la manipolazione di **query congiuntive**
- Classe per la manipolazione di **insiemi di viste**

SONO STATI AGGIUNTI METODI ALLE CLASSI DEL PACKAGE **DISDATASTRUCTURE** (PER ULTERIORI DETTAGLI SI RIMANDA AL JAVADOC - SEZIONE LINK)

30-12-2002 AVVIATO LO SVILUPPO DEI SEGUENTI PACKAGES

- **DISParser**: per l'acquisizione dei dati da file di testo e, successivamente, via XML (*Messineo e Ruzzi*)
- **DISLAVApproach**: per l'implementazione del rispettivo algoritmo (*De Nigris*)
- **DISLavGavCompile**: per l'implementazione dell' algoritmo di compilazione Lav-Gav(*Messineo*)

[\[...back to Top!\]](#)

- italiano....

A cura di: De Nigris Saverio, Messineo Gabriele e Ruzzi Marco

Supervisori: Rosati Riccardo, Cali Andrea e Lembo Domenico

Webmaster Ruzzi Marco

- implementation in java, big OO approach
- documentation mostly javadoc only
- prototype never published ... dead?

what happened?

- data mapping! here: ID mapping only.
- no integration with real databases.
- where's the theory...

other implementations:

very many, most left prototypic

- TSIMMIS, garlic, etc
- disatdis, dwq, infomix, etc...
- COIN, etc...
- AMOS II

- TSIMMIS, garlic, etc

The Stanford IBM Manager of Multiple Information Sources

tsimmis last 1998, o/w not free...

- stanford collection of data management projects

www-db.stanford.edu

- current projects look more at distributed databases

Übersicht Schema Anpassung

- disatdis, dwq, infomix, etc...
- unfinished, prototypic, EU funded...
- selfhosting projects

<http://www.dis.uniroma1.it/~disatdis/>

<http://www.dbnet.ece.ntua.gr/~dwq/>

<http://sv.mat.unical.it/infomix/>

(dead page!)

infomix: sv.mat.unical.it, kr.tuwien.ac.at,
rodan.pl, dis.uniroma1.it, dbai.tuwien.ac.at

Übersicht Schema Anpassung

- Timeline ranges from month 0 to month 35; month 0 = April 2002

Work-package No	Workpackage title	Lead contractor	Person-months	Start month	End month	Deliverable No
WP1	Appraisal of current methods, trends and techniques in information integration	TUWIEN	6	0	3	D1.1, D1.2, D1.3
WP2	Specification of the INFOMIX architecture	UNICAL	12	2	6	D2.1, D2.2, D2.3
WP3	Specification of the Integration Information Model (IIM)	UNIROME1	27	3	17	D3.1, D3.2, D3.3
WP4	Specification of the Integration Engine	UNICAL	30	9	17	D4.1, D4.2, D4.3
WP5	Algorithms for query (re)formulation and optimization	UNIROME1	24	12	23	D5.1, D5.2, D5.3
WP6	Heterogeneous data acquisition and transformation framework	TUWIEN	12	12	20	D6.1, D6.2
WP7	INFOMIX prototype	RODAN	42,5	18	31	D7.1, D7.2, D7.3, D7.4, D7.5
WP8	Assessment and dissemination	TUWIEN	9	0	35	D8.1, D8.2, D8.3, D8.4, D8.5, D8.6
WP9	Project management	UNICAL	9	0	35	D9.1, D9.2

<http://www.dis.uniroma1.it/~rosati/infomix/workpackages.htm>

more about logics....

COIN - COntext INterchange

<http://context.mit.edu/~coin/>

- prolog based algorithms
- many derived projects
- again, knowledge base AI, declarative systems, "aggregators"

--> wealth of algorithms

more practical:

- **AMOS II** <http://www.dis.uu.se/~udbl/amos/>
active mediators object system
- "prototype object-relational DMBS"

[...] AMOS II is a distributed mediator system [Wie92] [...] AMOS II mediators contain one or several wrappers [...] The declarative multi-database query language AMOSQL requires queries to be optimized before execution. The query compiler translates AMOSQL statements into object calculus and algebra expressions in an internal simple logic based language called ObjectLog [LR92], which is an OO dialect of Datalog [Ull88]. [...] The query optimizer is extensible through a generalized foreign function mechanism, [...] carefully optimized [...] AMOS II runs under Windows NT. The system uses around 350KB of code and 1500KB of meta data. [...] The AMOS II schema also contains descriptions of the properties of the wrapper.

Übersicht Schema Anpassung

in other words:

- is query rewriting == schema mapping?
- wrapper execution == data mapping?
- distribution vs heterogeneity
- materialized views as query cache?

- separated mediator execution engine?

what's left?

- real data! real heterogeneity!
- integration into working pipeline!
- analyzing schemas and seeking real correspondences or problems
- no virtual extract api, generate plain sql for source and target. Incl. bulk.
- no manual wrappers and foreign functions, generate code for usual conversions, e.g. arithmetic, collisions, id mapping

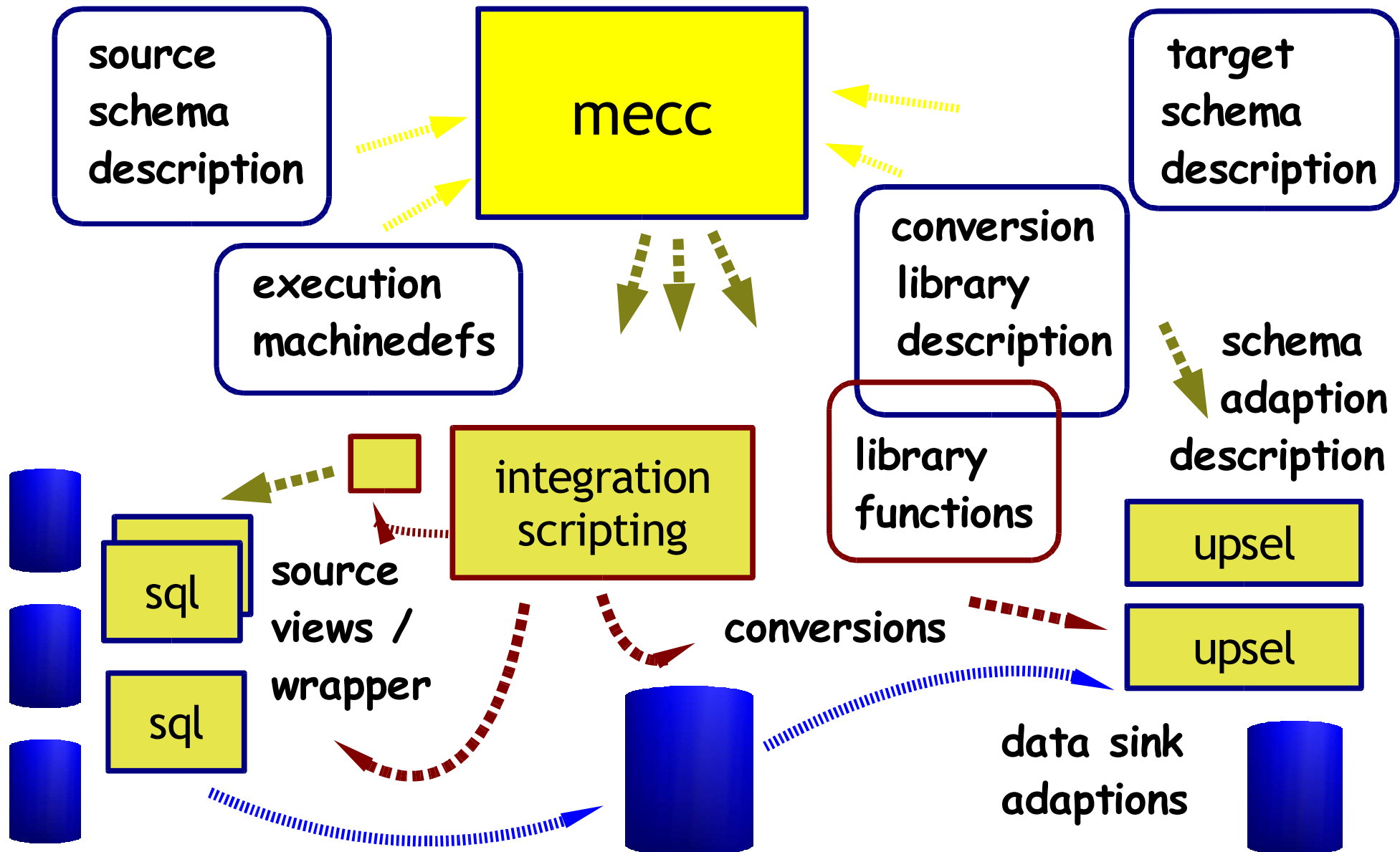
compiler-compiler approach

- analyzing extensional schema
correspondences
- generating functions, query views, query
rewriters by help of function libraries
- handover to heterogenic set of executors,
sql/views on source, mediator, target
DMBS, o/w some functional script engines
- description of pipeline processes to
achieve schema mapping execution.

integration in local research:

- columba, functional scripting in python?
- */mac, MTh research on data mapping
- generate sql/views for ETL in columba
- analyze schema correspondance and seek out possible problems (no need to guess)
- compare, try to find bulk loads, possible intermediate materialized views
- describe and weave functions written, (generate after */mac MTh results)

compiler-compiler overview:



python scripting

- used in columba, known to developers
- xml, vfs, db api readily available
- biopython function library
- massive retroinspection, live script generation and execution
- low platform project overhead

minus: sql/datalog algorithms (write them!)

speed of live query mapping (needed?)