Data Oriented Middleware

The Internet meets Databases

A presentation by

Yannis Papakonstantinou

on joint works with

Alin Deutsch,

Yannis Katsis,

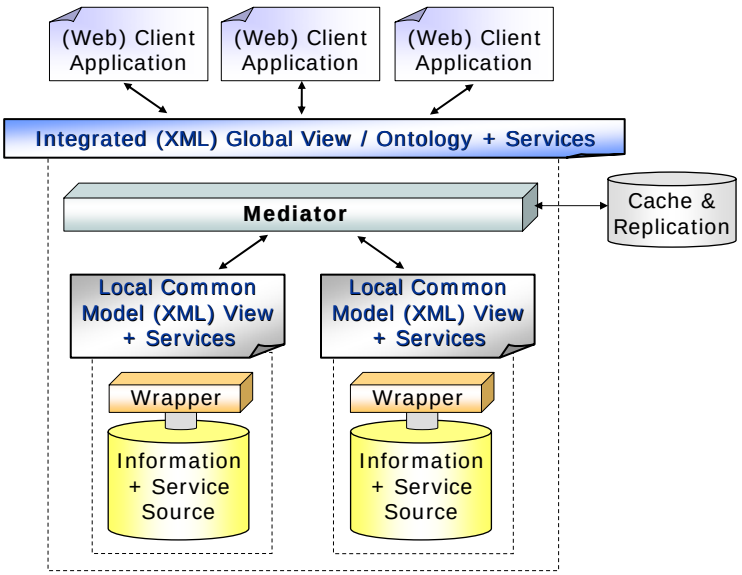
Michalis Petropoulos,

Vasilis Vassalos

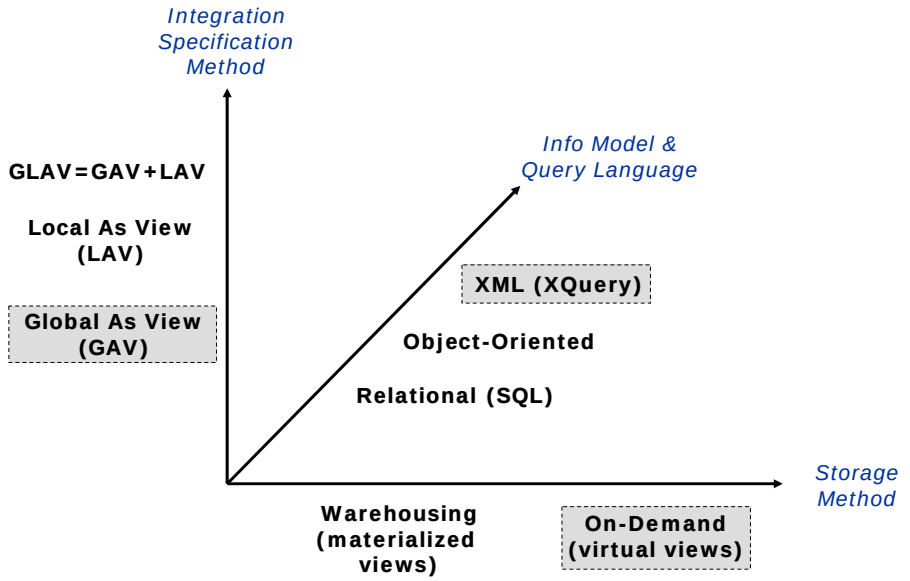
UCSD

CSE Department

Conventional Architecture for Unified Access to Data & Services



Approaches towards View-Based Data Integration



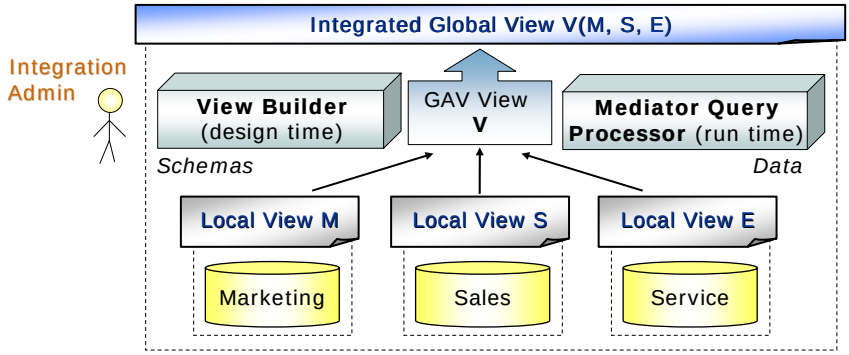
Industrial Involvement w/ Integration

- Enosys Software in 2000
  - Enosys = greek for “union, merge, fusion”
- Enterprise Information Integration product
  - XML, virtual, GAV view
- Product also sold under BEA’s Liquid Data brand since 2002
  - Acquired by BEA in 2003

Current Enterprise Information Integration Deployments

ENTERPRISE

- Small Domain
- Mostly Vertical Partition of Sources
- Primarily Application-Driven View or Identity View
- Integration Administrator/Developer in charge

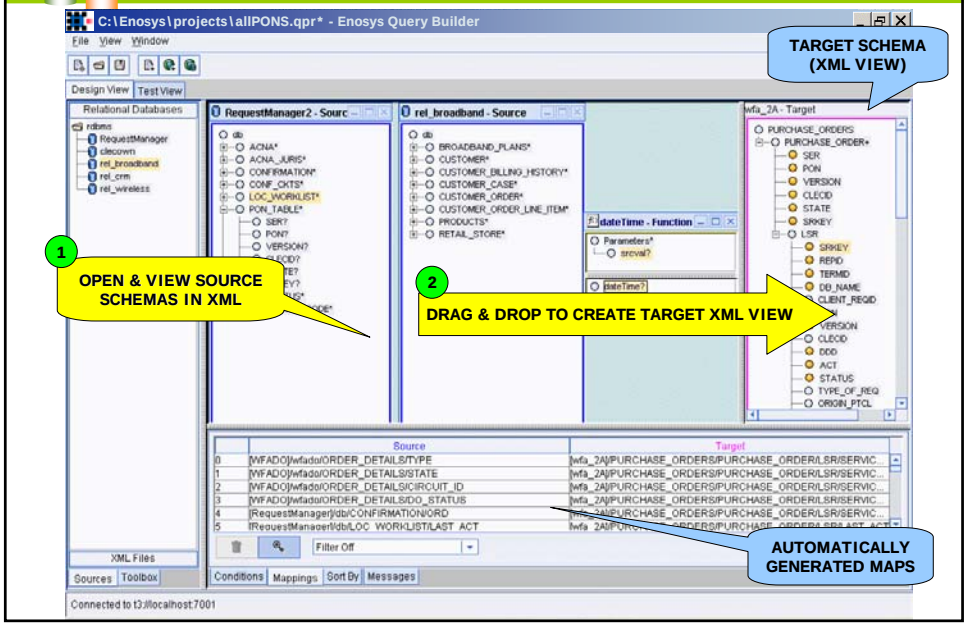


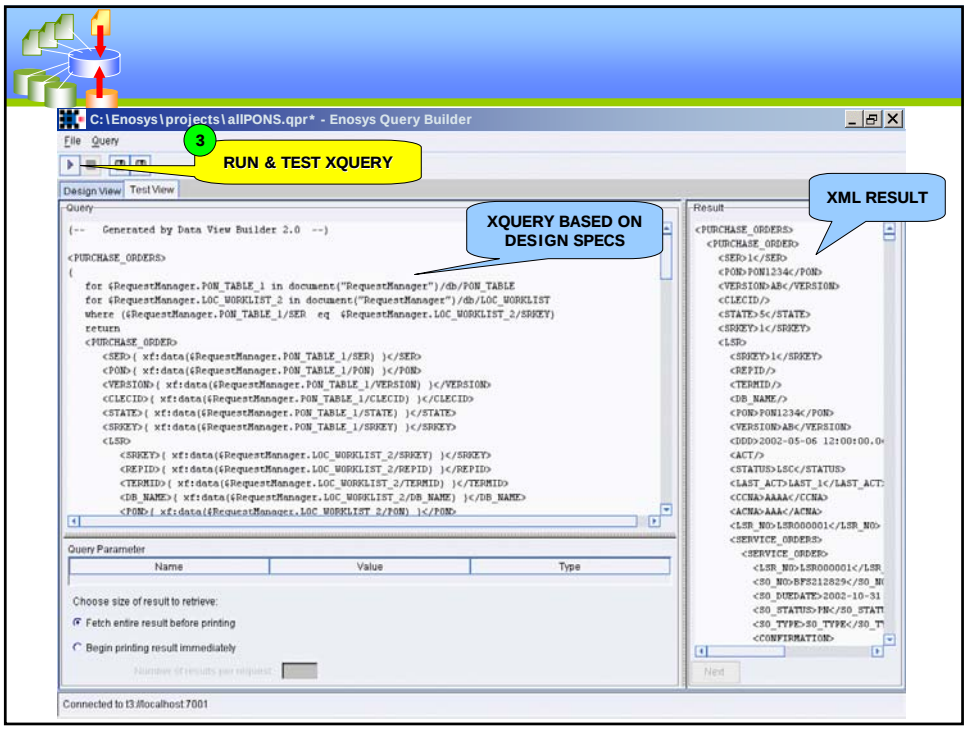
Opportunities and Needs Presented by "Motivated" Communities

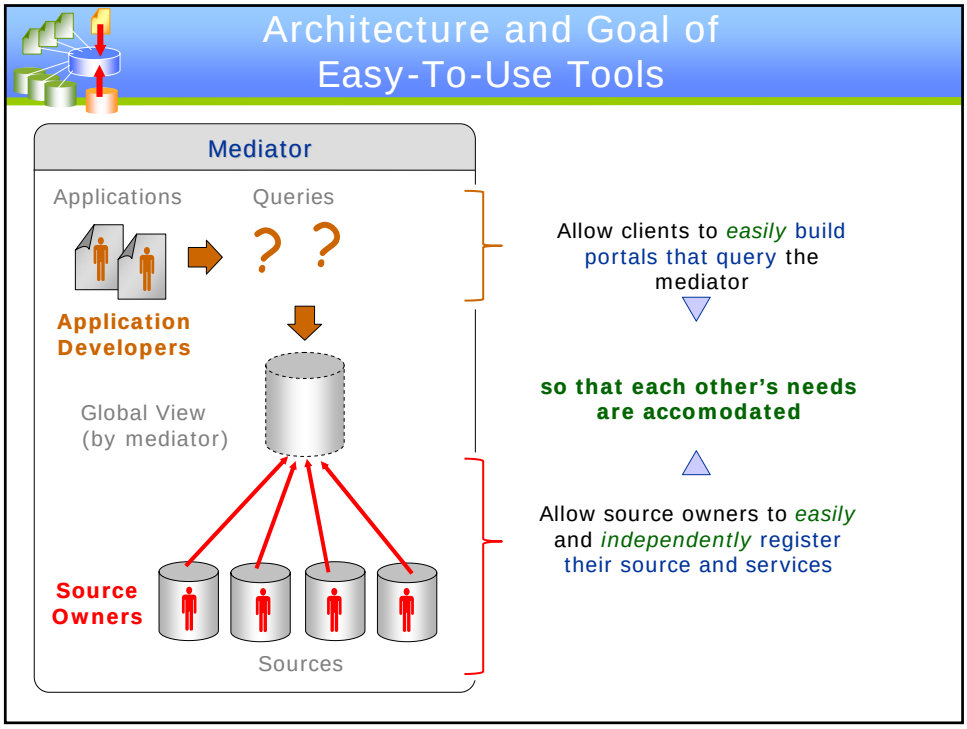
Communities

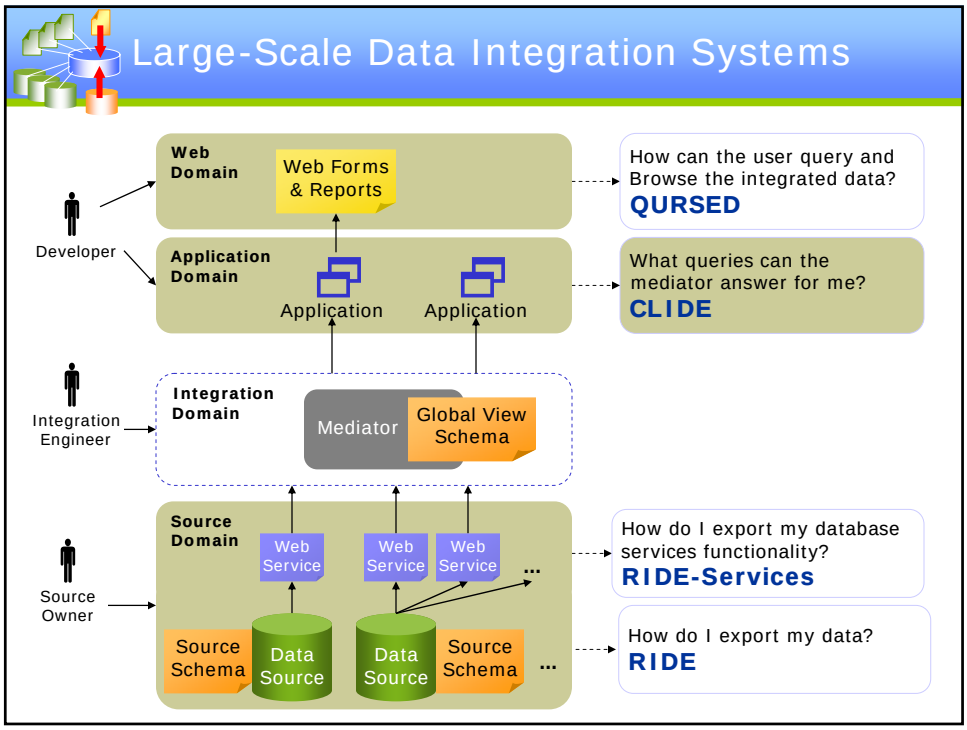
- Emerging Myriads of Internet Communities of
  - Myriads of sources and clients
  - Source owners motivated to participate
- EII does not address needs
  - Expensive
  - Bottleneck of Single Integration Admin
- Make building corresponding portals as easy as starting and participating in newsgroups
- Appropriate tools needed to enable source owner and client participation

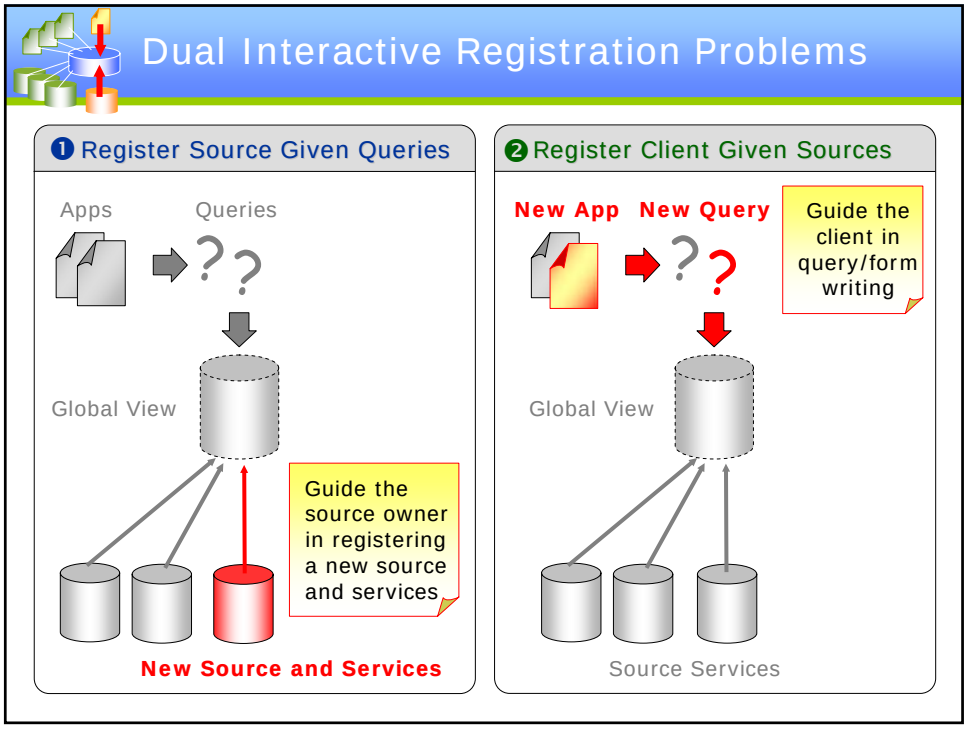
Visual Tools Matter!
(example from the Enosys Query Builder)

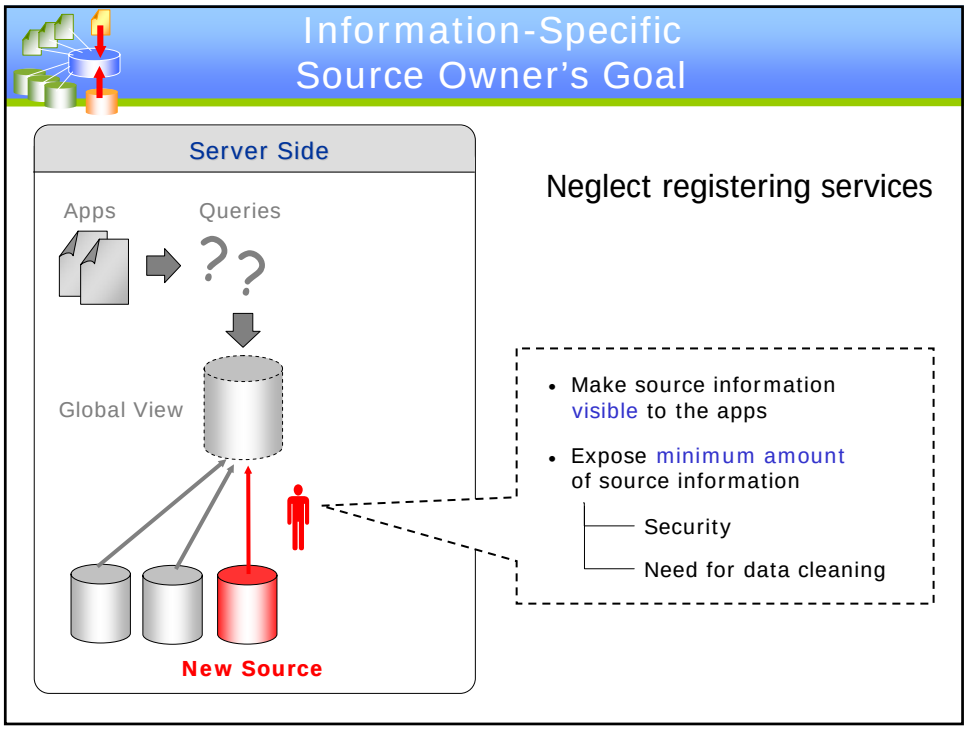




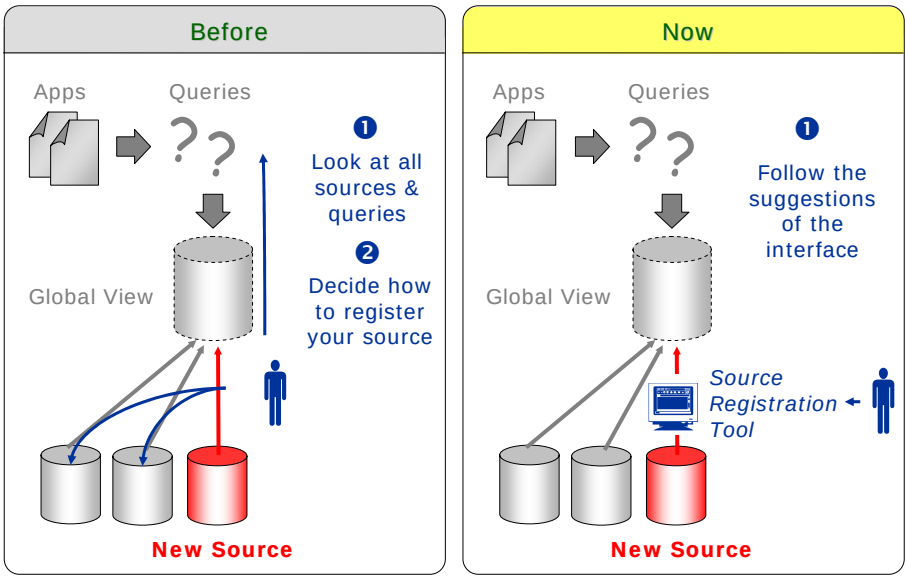

Architecture and Goal of
Easy-To-Use Tools
Mediator
Applications
Queries
Application Developers
Global View (by mediator)
Source Owners
Sources
Allow clients to easily build portals that query the mediator
so that each other's needs are accomodated
Allow source owners to easily and independently register their source and services


Large-Scale Data Integration Systems
Web Domain
Web Forms & Reports
Developer
Application Domain
Application
Application
Integration Engineer
Integration Domain
Mediator
Global View Schema
Source Owner
Source Domain
Web Service
Web Service
Web Service
...
Source Schema
Data Source
Data Source
Source Schema
...
How can the user query and Browse the integrated data?
QURSED
What queries can the mediator answer for me?
CLIDE
How do I export my database services functionality?
RIDE-Services
How do I export my data?
RIDE


Dual Interactive Registration Problems
1 Register Source Given Queries
Apps
Queries
Global View
Guide the source owner in registering a new source and services
New Source and Services
2 Register Client Given Sources
New App
New Query
Guide the client in query/form writing
Global View
Source Services


Information-Specific
Source Owner's Goal
Server Side
Apps
Queries
Global View
New Source
Neglect registering services
Make source information visible to the apps
Expose minimum amount of source information
Security
Need for data cleaning

How to achieve this Goal



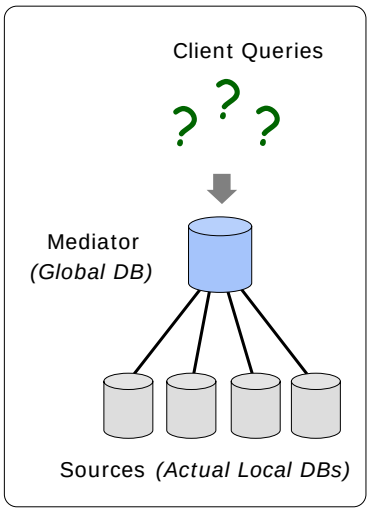
Our Goal in Source Registration

Guide the source owner visually through the **registration** of the **source**

so as to **contribute information** to the **answer of the queries**

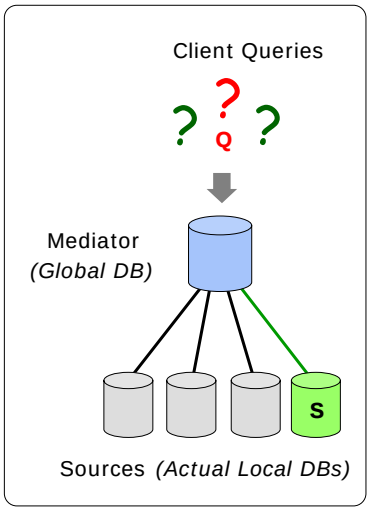
while exposing the minimum information possible and/or minimizing effort

The Problem



15

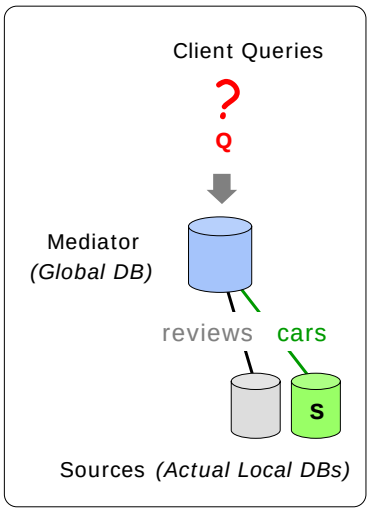
The Problem



- What is the contribution of source S to the result of the query Q?

16

The Problem



- What is the contribution of source S to the result of the query Q?

Q: cars

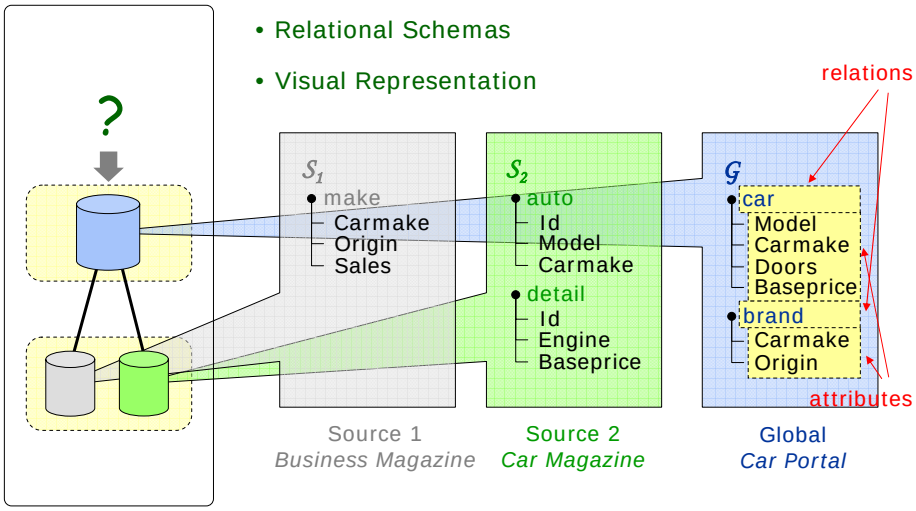
S is Self Sufficient w.r.t. Q

Q: cars JOIN reviews

S is Now Complementary w.r.t. Q

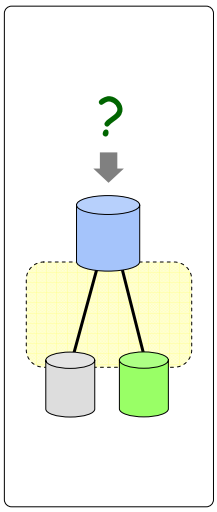
17

Relational Schemas: Local and Global



18

Source Registration using GLAV Mappings



- Source Registration:
  Correspondence between a source schema and the global schema
  =
  Set of Mapping Constraints of the form

$(U \subseteq V)$

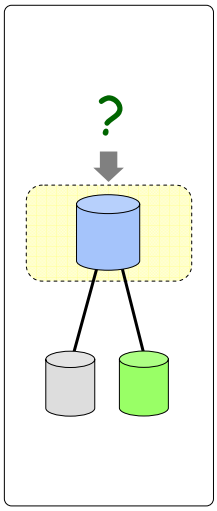
$CQ^{=}$ over source schema

$CQ^{=}$ over global schema

- Open World
- Global and Local As View (GLAV)

19

Target Constraints



- Constraints on the global schema
  =
  Set of Constraints of the form

$(U \subseteq V)$

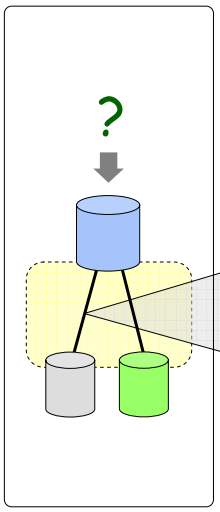
$CQ^{=}$ over global schema

$CQ^{=}$ over global schema

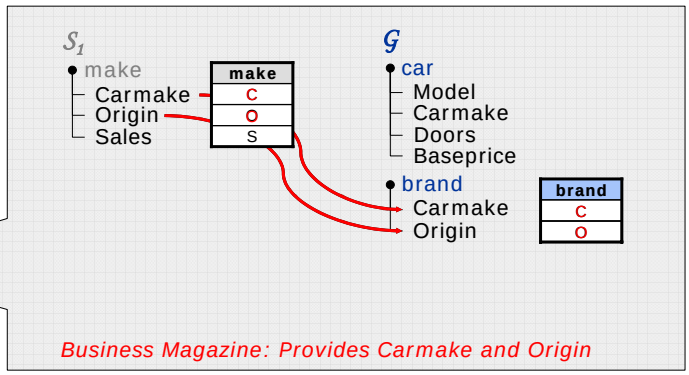
- Also Expresses Dependencies (PKs, Ref Integrity, ...)

20

Visual Representation of Mappings (1)



- Visual Representation (IBM Clio)

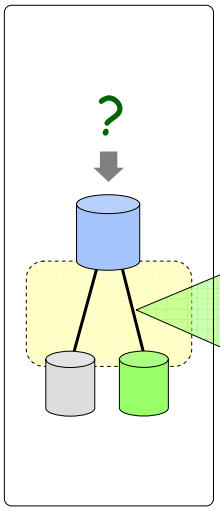


$(U_1 \subseteq V_1)$

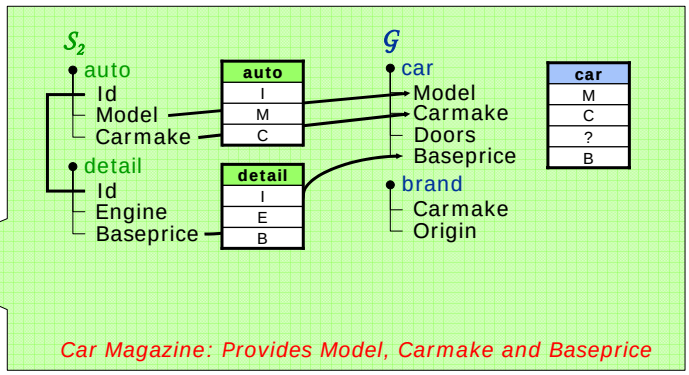
$U_1(C, O)$:- make(C, O, S)
$V_1(C, O)$:- brand(C, O)

21

Visual Representation of Mappings (2)

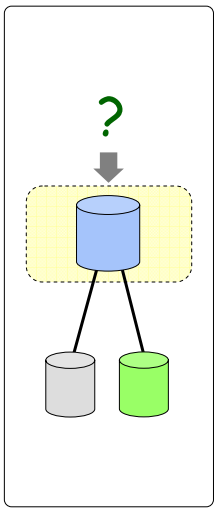


- Visual Representation (IBM Clio)



22

Example of Target Constraint



- (Model, Carmake) is a PK of car

$\mathcal{G}$
- car
  - Model
  - Carmake
  - Doors
  - Baseprice
- brand
  - Carmake
  - Origin

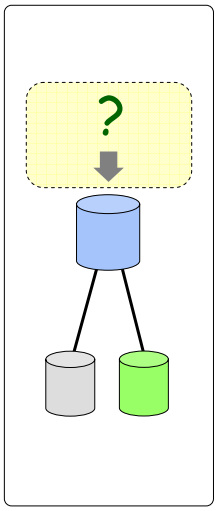
$(U_1 \subseteq V_1)$

$U_1(M, C, D_1, B_1, D_2, B_2)$:- $car(M, C, D_1, B_1)$, $car(M, C, D_2, B_2)$

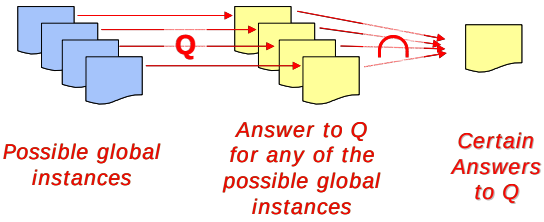
$V_1(M, C, D, B, D, B)$:- $car(M, C, D, B)$

23

Query Semantics



- Queries in $UCQ^{=}$
- Set of Possible Global Instances
  Set of global instances that satisfy all constraints
- Query Answers = Set of Certain Answers
  The tuples appearing in the answer to Q for any possible global instance



24

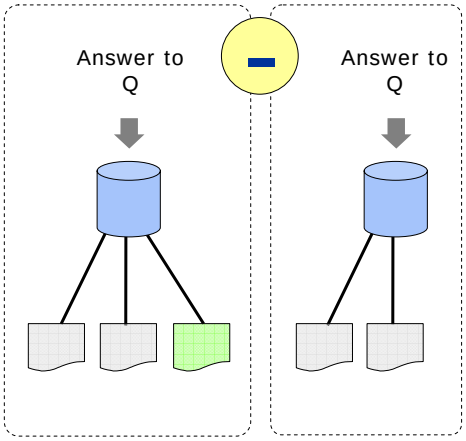
Source Instance's Contribution

- *For given instances of the sources*

Contribution to Q of Source Instance

=

The tuples in answer of Q not provided by the other sources

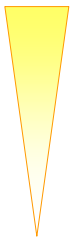


25

Source Registration's Contribution

- Source Registration: Source Mappings
- Degrees of Source Registration's Contribution

1. Self Sufficient
2. Now Complementary
3. Later Complementary
4. Unusable

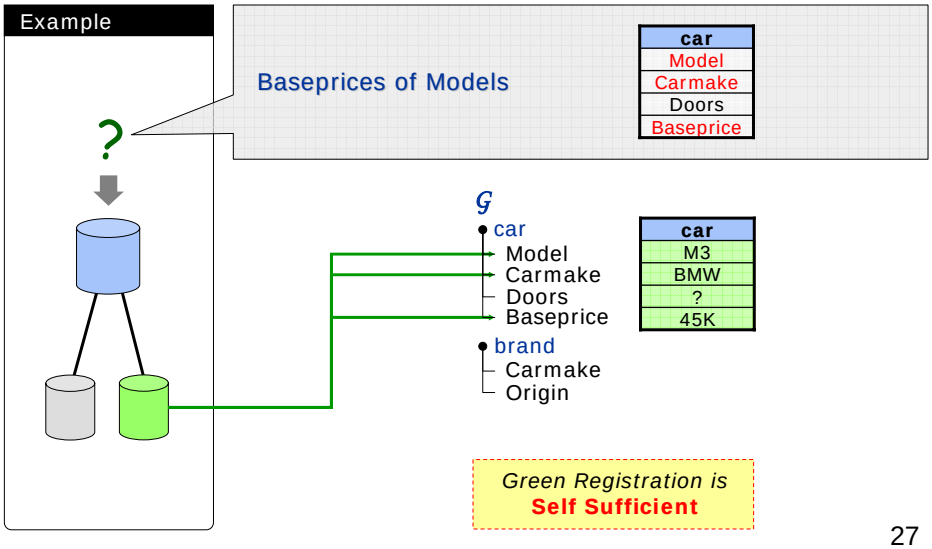


More contribution

Less contribution

26

Self Sufficient Registration: Example



27

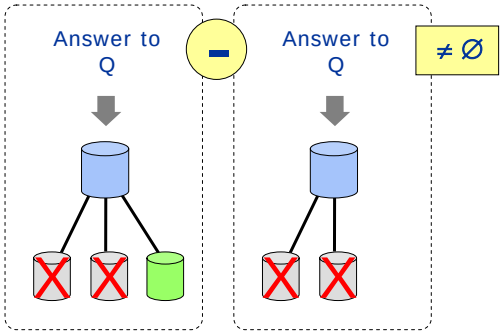
Self Sufficient Registration: Definition

❶ Self Sufficient

$\exists$ Source instance

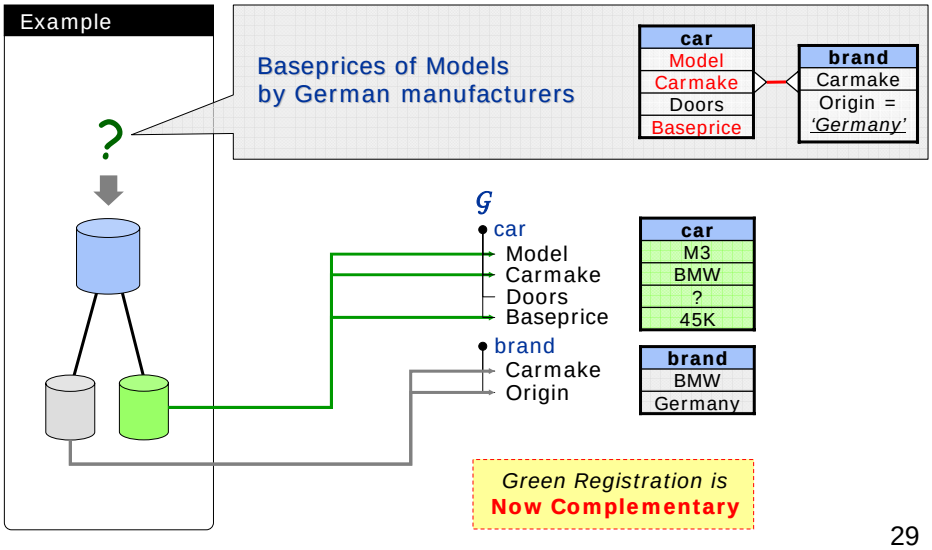
s.t.

The source has a non empty contribution in the absence of the other sources



28

Now Complementary Registration: Example



29

Now Complementary Registration: Definition

❷ Now Complementary

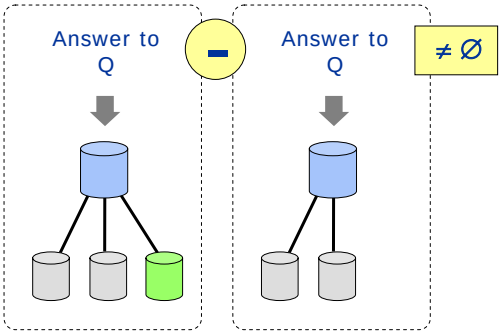
Not Self Sufficient

&

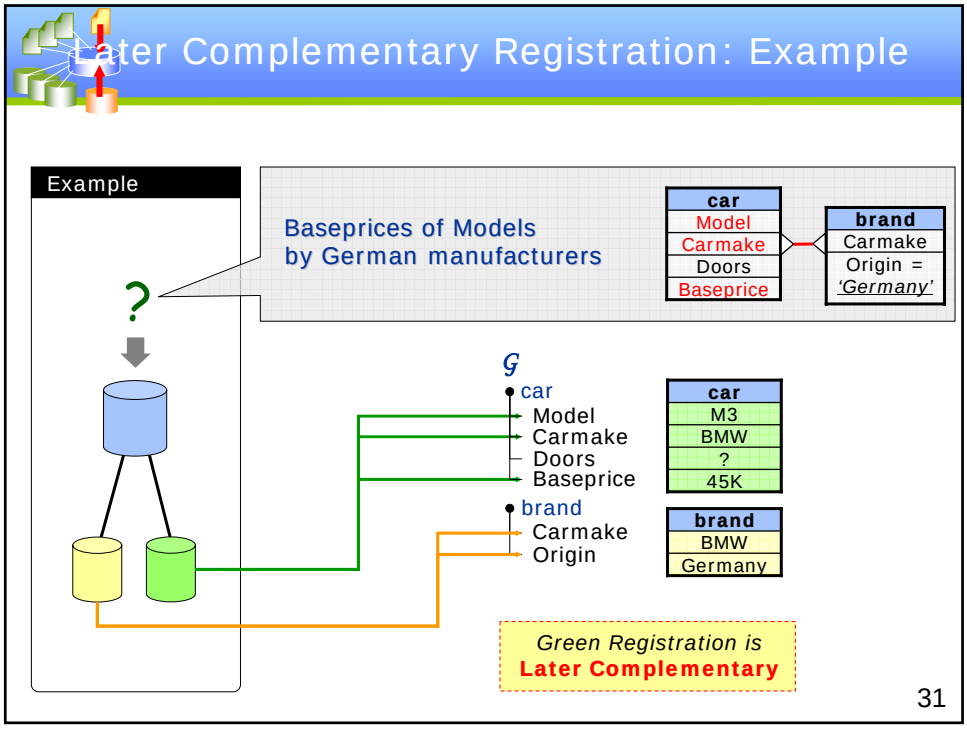
∃ Source instances

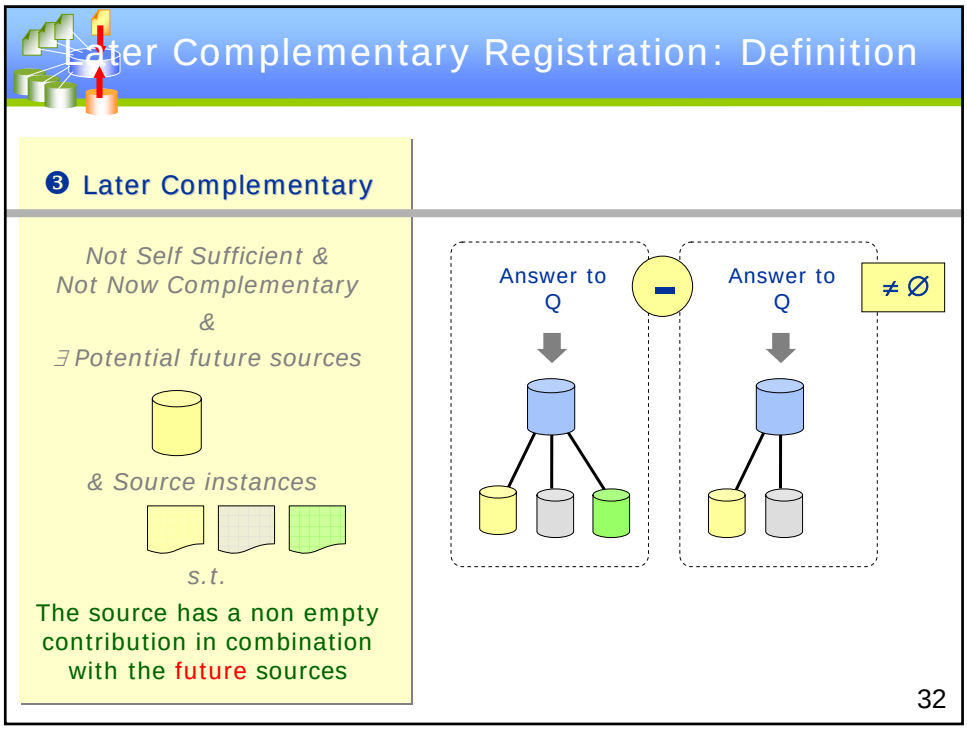
s.t.

The source has a non empty contribution in combination with the other existing sources

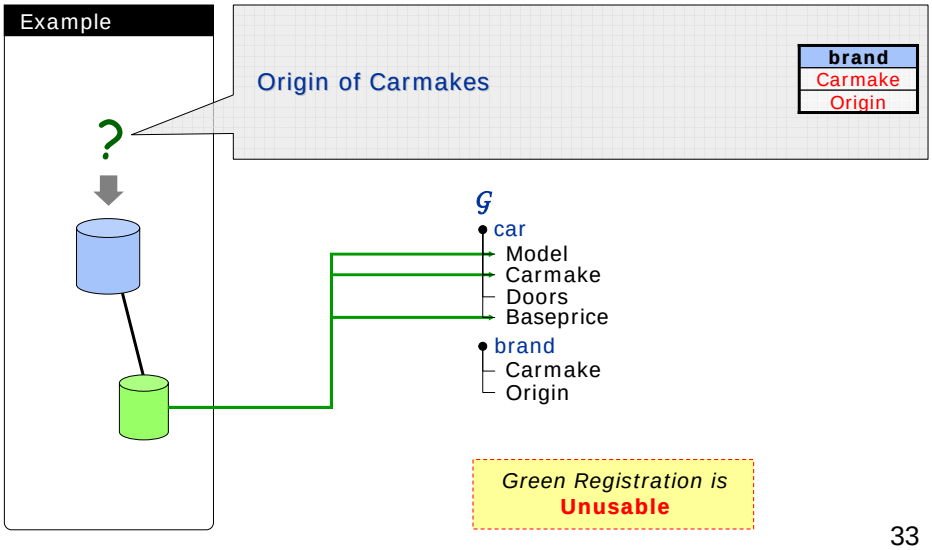


30


Later Complementary Registration: Example
Example
Baseprices of Models
by German manufacturers
car
Model
Carmake
Doors
Baseprice
brand
Carmake
Origin =
'Germany'
?
$\mathcal{G}$
car
Model
Carmake
Doors
Baseprice
brand
Carmake
Origin
car
M3
BMW
?
45K
brand
BMW
Germany
Green Registration is
Later Complementary
31


Later Complementary Registration: Definition
❸ Later Complementary
Not Self Sufficient &
Not Now Complementary
&
∃ Potential future sources
& Source instances
s.t.
The source has a non empty contribution in combination with the future sources
Answer to Q
−
Answer to Q
≠ ∅
32

Unusable Registration: Example



Green Registration is **Unusable**

33

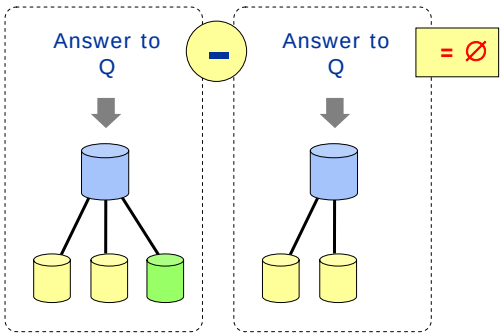
Unusable Registration: Definition

❹ Unusable

Not Self Sufficient &
Not Now Complementary &
Not Later Complementary

⇔

The source has a empty contribution regardless of what sources enter the system



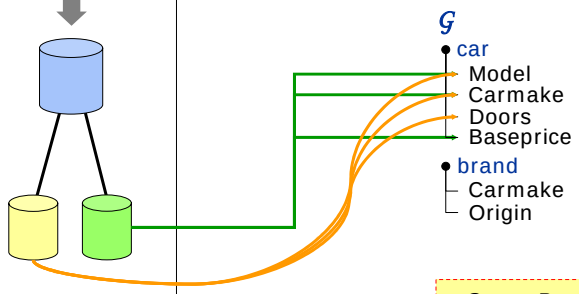
34

Subtleties for Unusable Registrations

Example

Baseprices and Doors of Models

car
Model
Carmake
Doors
Baseprice



car
M3
BMW
?
45K

car
M3
BMW
2
?

Green Registration is **Unusable**

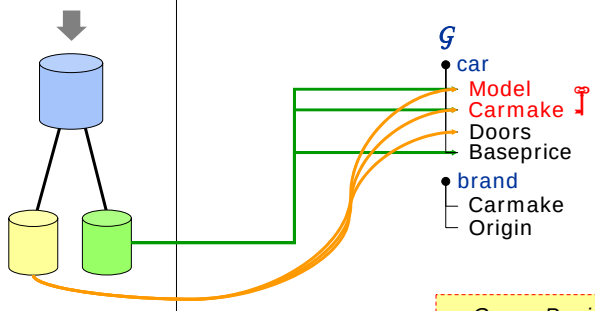
35

In presence of PK Unusable Example becomes Later Complementary

Example

Baseprices and Doors of Models

car
Model
Carmake
Doors
Baseprice



car
M3
BMW
2
45K

Green Registration is **Later Complementary**

36

Decidability Results

Overview: What is decidable

Degrees	Target constraints: None	Target constraints: Primary keys	Target constraints: Primary keys + Referential Integrity Constraints
Self Sufficient	Yes	Yes	No
Now complementary	Yes	Yes	No
Later complementary	Yes	Yes	?
Unusable	Yes	Yes	?

37

Issues

Unique client query **Vs** *Multiple* client queries ❶

Contribute to:
- all queries?
- one query?
- specific queries?
- some queries based on some ranking?

Data *independence* **Vs** Data *dependence* ❷

e.g. M_1: cars, refPrices
M_2: reviews
Q: cars JOIN reviews JOIN refPrices

DB_1: cars, refPrices (Audis)
DB_2: reviews (Hondas)

(M_2, Q) now-complementary
but Certain Answers for Instances DB_1, DB_2 = $\varnothing$

Putting it all together

Architecture

Query Answering / Mappings / Schemas

Contribution

4 categories:

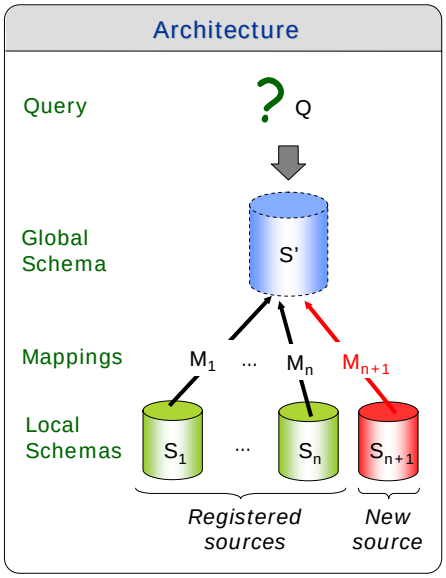
Self Sufficient / Now Complementary / Later Complementary / Unusable

Goal

Guide the source owner visually through the *registration* of the *source*

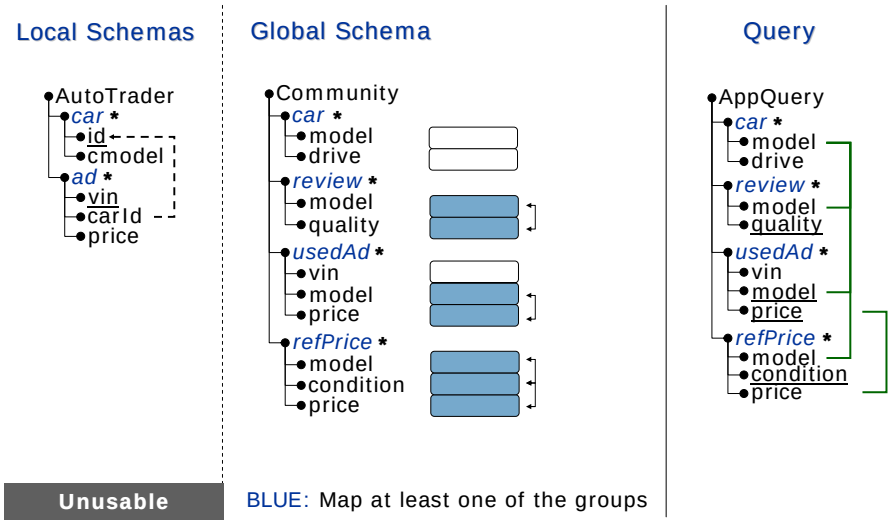
so as to *raise contribution* to the *answer of the queries*

while *exposing the minimum info possible* and/or minimizing effort



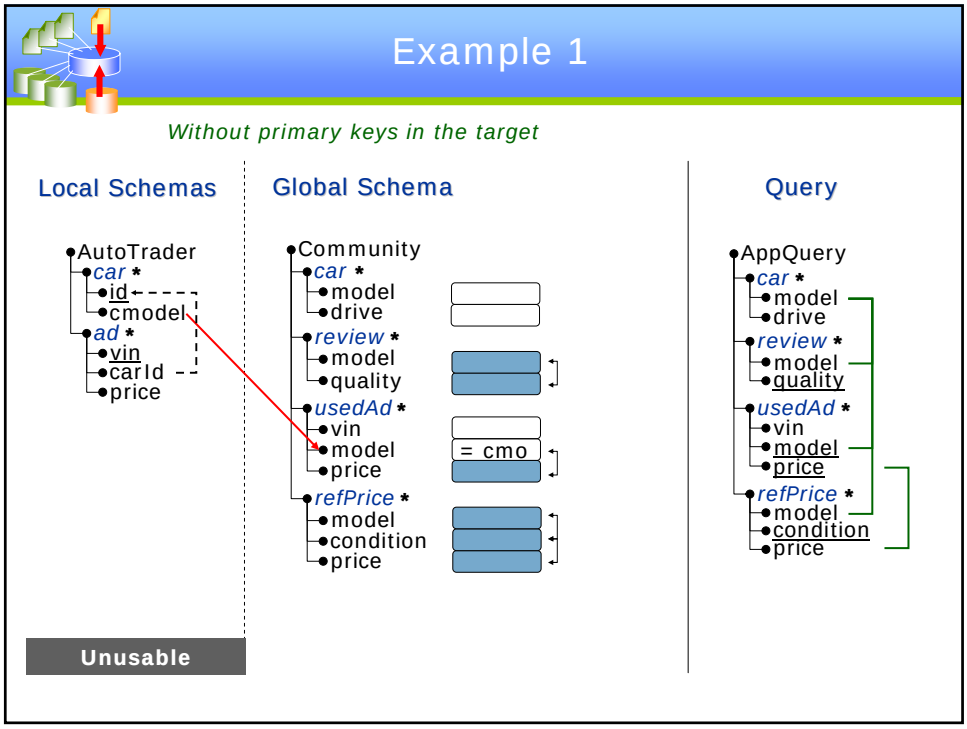
Example 1

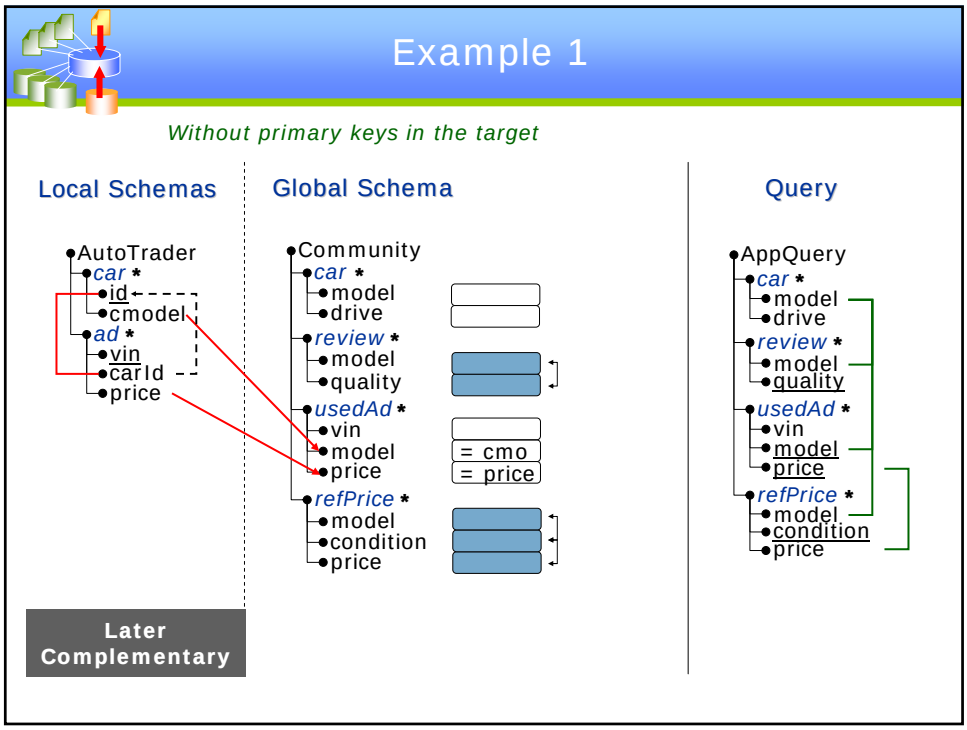
Without primary keys in the target

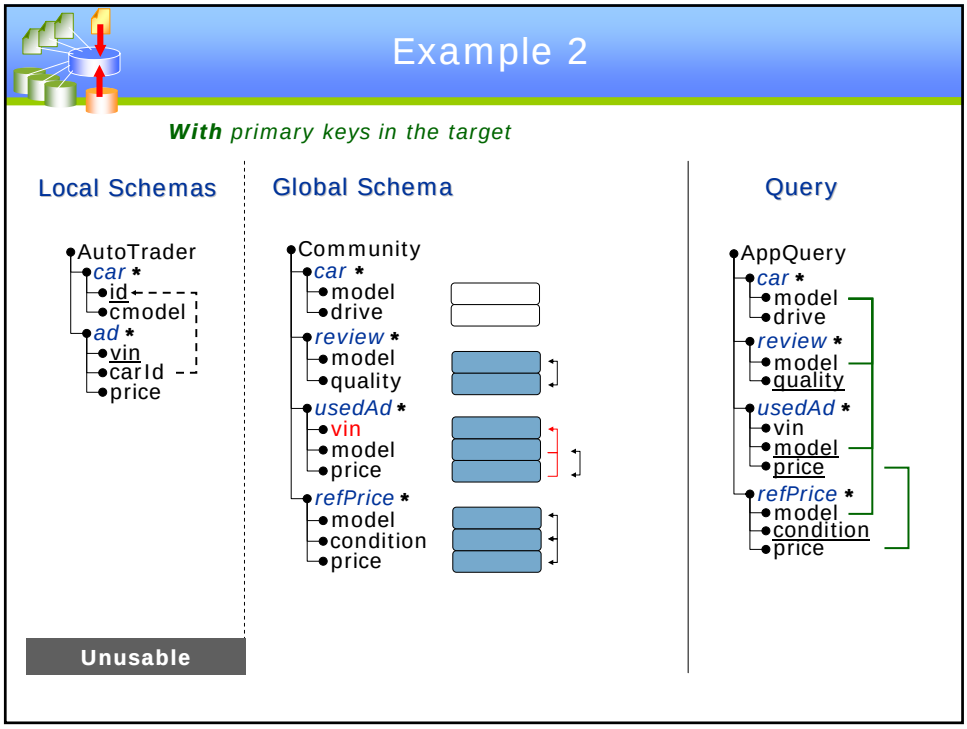


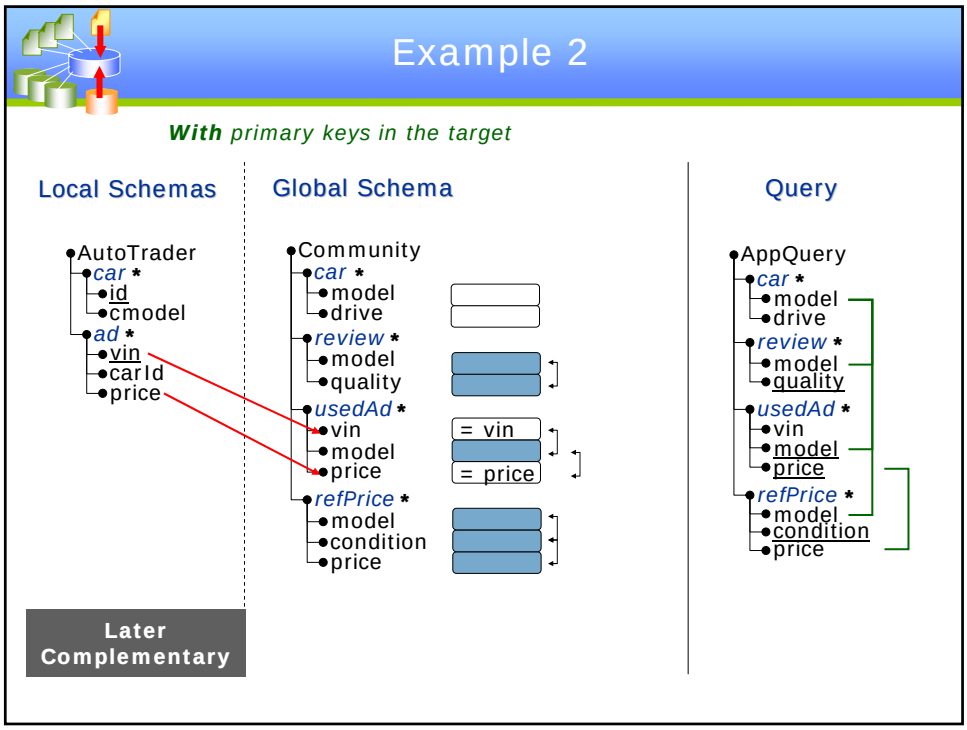
Unusable

BLUE: Map at least one of the groups


Example 1
Without primary keys in the target
Local Schemas
Global Schema
Query
AutoTrader
car *
id
cmodel
ad *
vin
carId
price
Community
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
= cmo
AppQuery
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
Unusable


Example 1
Without primary keys in the target
Local Schemas
Global Schema
Query
AutoTrader
car *
id
cmodel
ad *
vin
carId
price
Community
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
= cmo
= price
AppQuery
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
Later
Complementary


Example 2
With primary keys in the target
Local Schemas
Global Schema
Query
AutoTrader
car *
id
cmodel
ad *
vin
carId
price
Community
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
AppQuery
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
Unusable


Example 2
With primary keys in the target
Local Schemas
Global Schema
Query
AutoTrader
car *
id
cmodel
ad *
vin
carId
price
Community
car *
model
drive
review *
model
quality
usedAd *
vin
= vin
model
price
= price
refPrice *
model
condition
price
AppQuery
car *
model
drive
review *
model
quality
usedAd *
vin
model
price
refPrice *
model
condition
price
Later Complementary

Lessons learned

Target constraints make a difference

1. ***To merge data with that of other sources*** *(become complementary):*

   Pick a relation and provide…

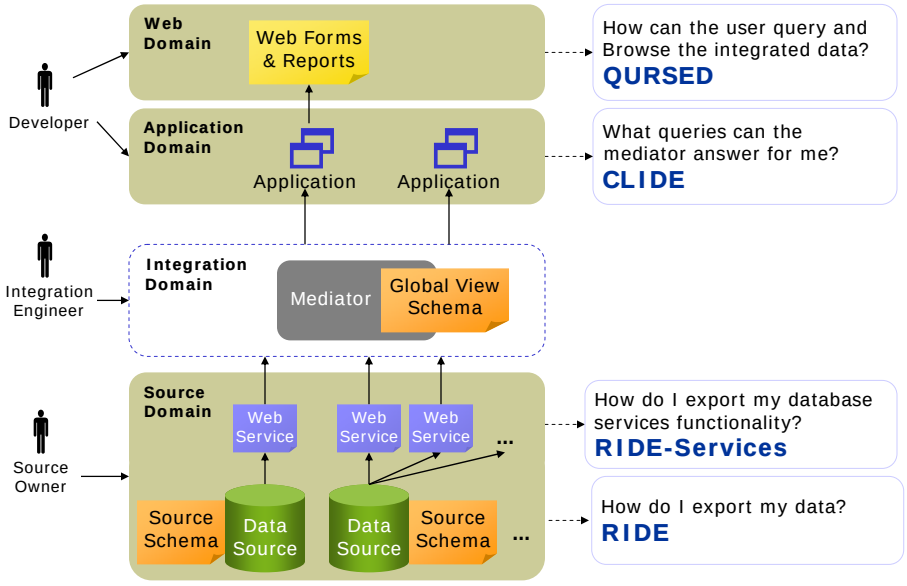
   In absence of primary keys
   …all its attributes asked by the query

   In presence of primary keys
   …its primary key and one of its attributes asked by the query

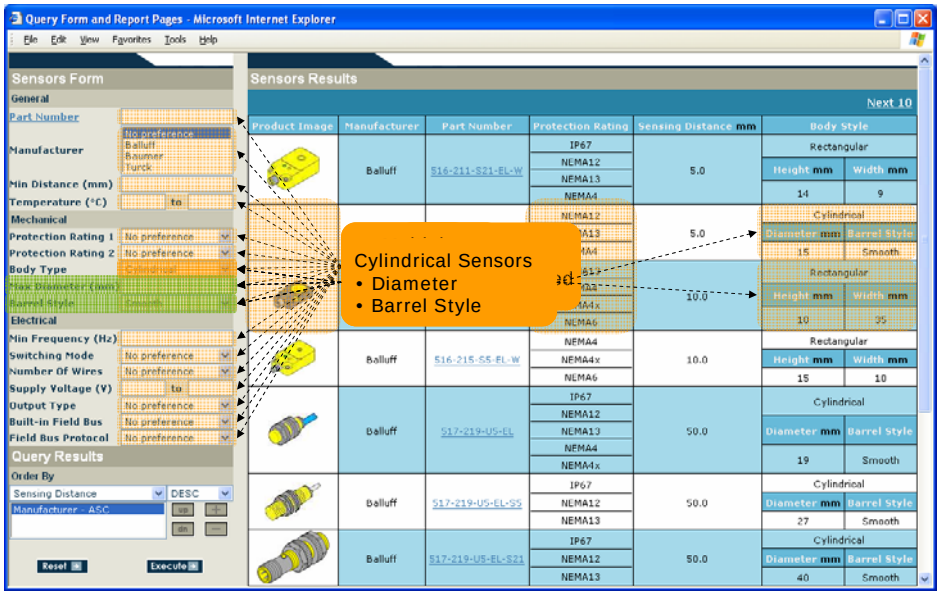
   The number of choices increases in presence of primary keys

2. ***Foreign keys on the target affect the suggestions***

Large-Scale Data Integration Systems



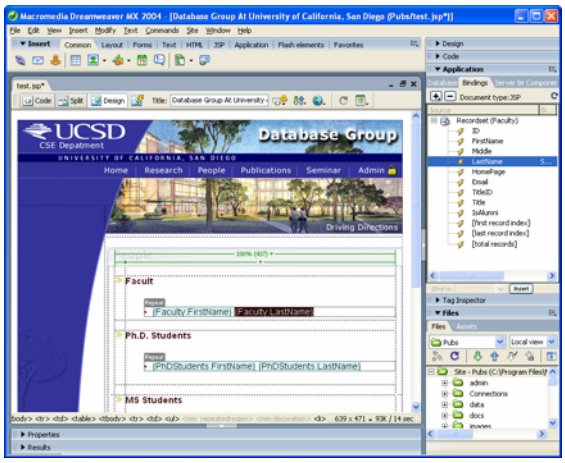
Challenge: Heterogeneity



Query Forms and Reports

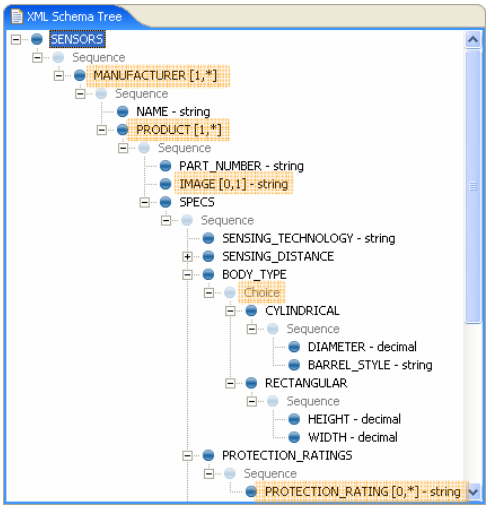
Current Practices

Macromedia



- Fixed Parameterized SQL Statements
- Custom Code Required for Combinations of Conditions
- Very Tricky for Semistructured Data

Capturing Heterogeneity



XML Schema

- Captures heterogeneity

XML

- Just a language
- Not a program

XQuery Standard (W3C)

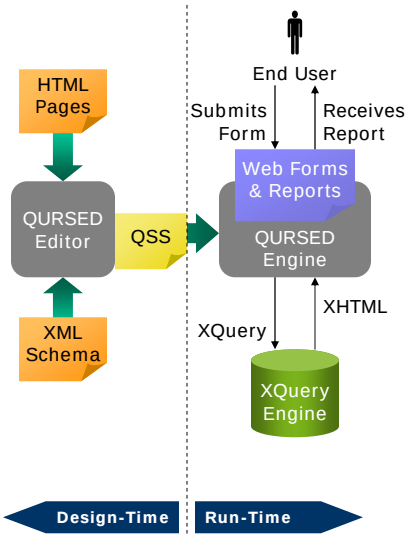
- XML Query Language
- Behind the scenes

XQuery: Expressive But Verbose

```
<html>
  <body>
    <table>{
    FOR $doc IN accessSource('XMLDB'), $S IN $doc/sensors,
        $MAN IN $S/manufacturer, $PROD IN $MAN/product,
        $SPEC IN $PROD/specs, $PROTS IN $SPEC/protection_ratings,
        $PROT1 IN $PROTS/protection_rating, $PART IN
     $PROD/part_number,
        $DIST IN $SPEC/sensing_distance, $BODY IN
     $SPEC/body_type,
        $N_BODY IN $BODY/name(), $PROT IN
     $PROTS/protection_rating,
        $NAME IN $MAN/name
    WHERE
        $PROT1 = "NEMA3"
        AND ((SOME $CYL IN $BODY/cylindrical,
                   $DIA IN $CYL/diameter,
                   $BAR IN $CYL/barrel_style
              SATISFIES
                   $DIA <= 20 AND $DIA <= 40)
             OR
             (SOME $REC IN $BODY/rectangular,
                   $HEI IN $REC/height,
                   $WID IN $REC/width
              SATISFIES
                   $HEI <= 20 AND $WID <= 40))
    ORDER BY $NAME DESCENDING, $DIST
    RETURN
        GROUPBY $PROD, $NAME, $DIST AS
        <tr>{
            <td>{
            FOR $IMG IN $PROD/image
            RETURN
                GROUPBY $IMG AS <img srv="{$IMG}" />
            }</td>,
            <td>{
                IF ($NAME = "Turck") THEN <img="turck.gif"</img>
                IF ($NAME = "Balluff") THEN
     <img="balluff.gif"</img>
                IF ($NAME = "Baumer") THEN
     <img="baumer.gif"</img>
            }</td>,
            {
```

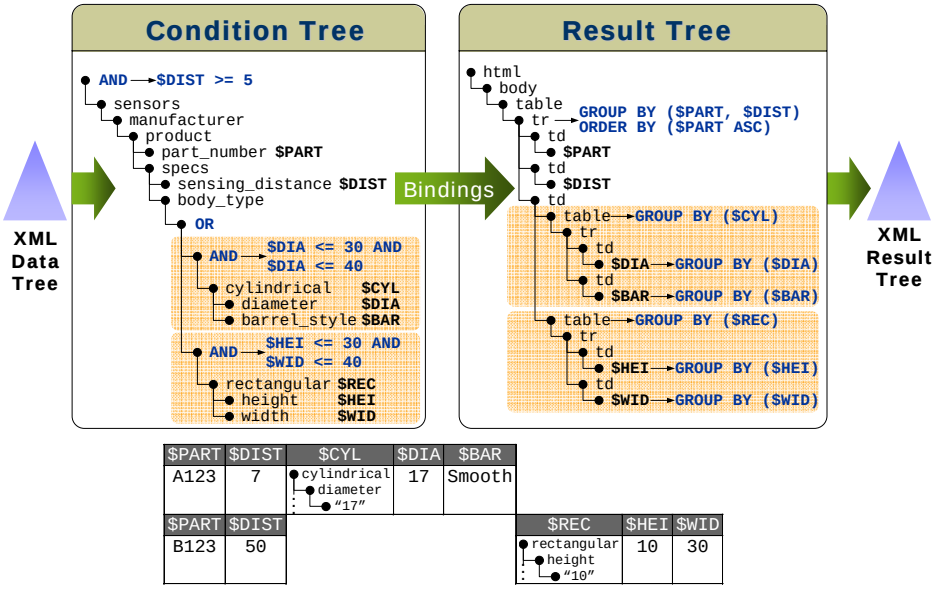
```
<td>{
    <table>{
        <tr>{
            GROUPBY $N_BODY AS <td>{$N_BODY}</td>
        }</tr>,
        <tr>{
            <td>{
            FOR $CYL IN $BODY/cylindrical,
                $DIA IN $CYL/diameter,
                $BAR IN $CYL/barrel_style
            WHERE
                $DIA <= 20 AND $DIA <= 40
            RETURN
                GROUPBY $CYL AS
                <table>{
                    <tr>{
                        GROUPBY $DIA AS
<td>{$DIA}</td>,
                        GROUPBY $BAR AS
<td>{$BAR}</td>
                    }</tr>
                }</table>
            }</td>,
            <td>{
            FOR $REC IN $BODY/rectangular,
                $HEI IN $REC/height,
                $WID IN $REC/width
            WHERE
                $HEI <= 20 AND $WID <= 40
            RETURN
                GROUPBY $REC AS
                <table>{
                    <tr>{
                        GROUPBY $HEI AS
<td>{$HEI}</td>,
                        GROUPBY $WID AS
<td>{$WID}</td>
                    }</tr>
                }</table>
            }</td>
        }</tr>
    }</table>
}</td>
```

The QURSED Solution

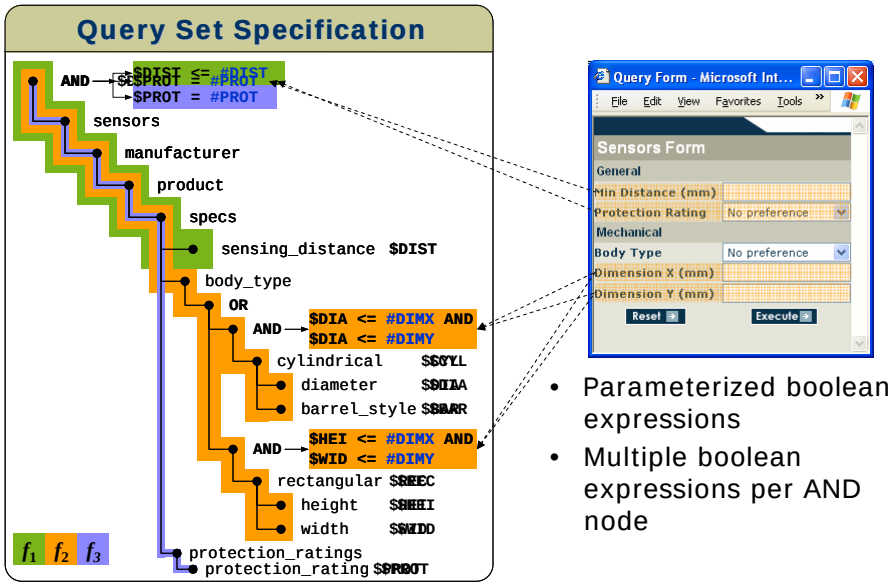


- XML Schema-Driven
- Declarative!
  - Separation of querying & presentation
  - Model-View-Controller pattern
- No programming required!
  - Editor provides visual actions
  - Automatic report construction
- Query Set Specification (QSS)

Tree Query Language (TQL)

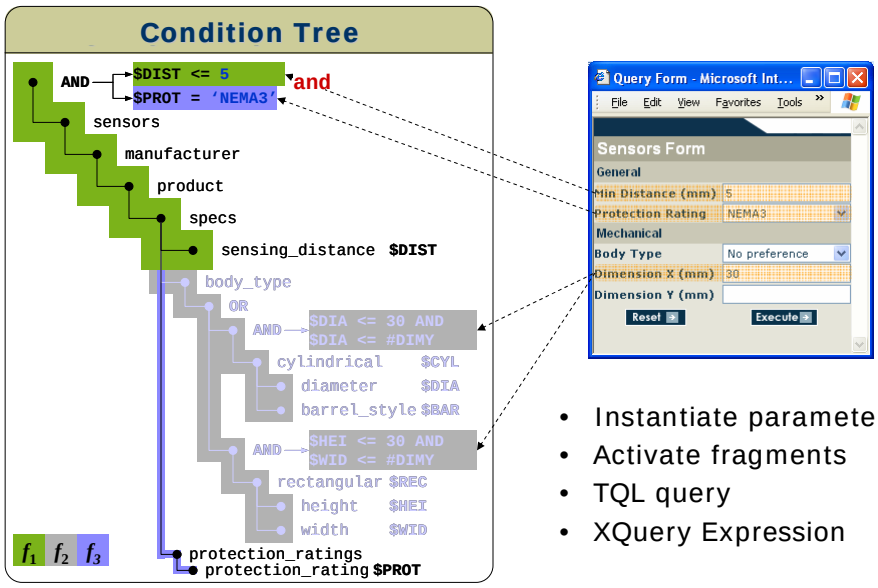


Query Set Specification (QSS): Addresses Condition Combinations



- Parameterized boolean expressions
- Multiple boolean expressions per AND node

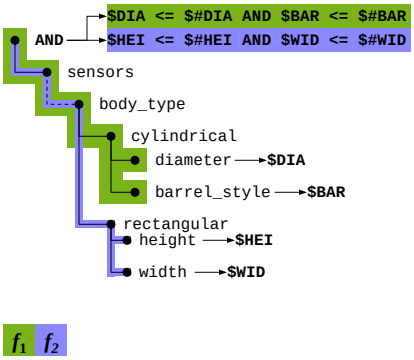
Run-Time: QSS to TQL Queries

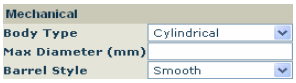


- Instantiate parameters
- Activate fragments
- TQL query
- XQuery Expression

Fragment Dependencies Address Heterogeneity

Meaningful Queries

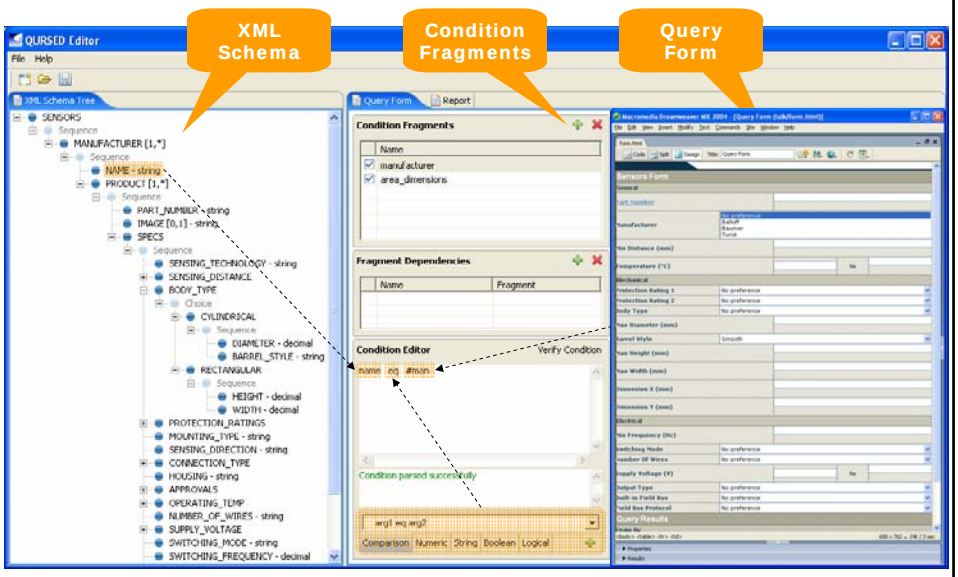




<f1, #BODY = 'Cylindrical'>

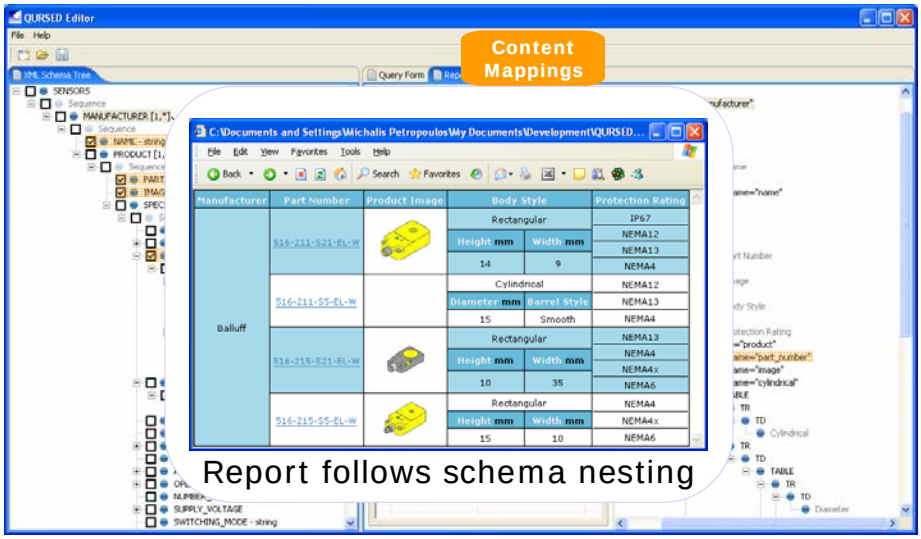
<f2, #BODY = 'Rectangular'>

QURSED Editor: No Coding

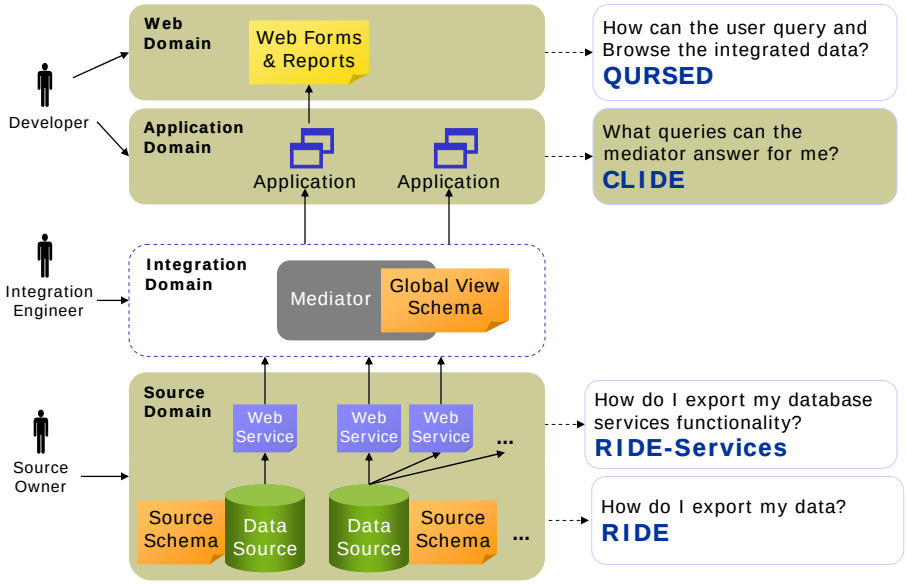


QURSED Editor: Automatic Reports

Building Reports



Large-Scale Data Integration Systems



Running Example

Parameterized Views

Dell

Schema

Computers(cid, cpu, ram, price)
NetCards(cid, rate, standard, interface)

Views

Computers for a given ***cpu***

V1 ***ComByCpu(cpu) → (Computer)****
SELECT DISTINCT Com1.*
FROM Computers Com1
WHERE Com1.cpu=***cpu***

V2 ***ComNetByCpuRate(cpu, rate) → (Computer, NetCard)****
SELECT DISTINCT Com1.*, Net1.*
FROM Computers Com1, Network Net1
WHERE Com1.cid=Net1.cid
AND Com1.cpu=***cpu***
AND Net1.rate=***rate***

Computers & **NetCards** for a given ***cpu*** & ***rate***

Cisco

Schema

Routers(rate, standard, price, type)

Views

***Wired* Routers**

V3 ***RouByTypeW() → (Router)****

SELECT DISTINCT Rou1.*
FROM Routers Rou1
WHERE Rou1.type='**Wired**'

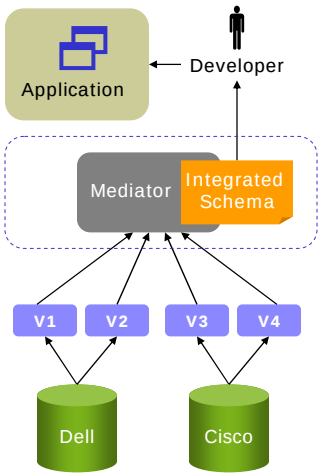
***Wireless* Routers**

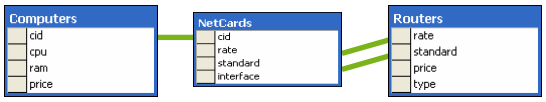
V4 ***RouByTypeWL() → (Router)****

SELECT DISTINCT Rou1.*
FROM Routers Rou1
WHERE Rou1.type='**Wireless**'

Running Example

Integrated Schema





- Integrated schema puts together the Dell and Cisco schemas

Attribute Associations

- (Computers.cid, NetCards.cid)
- (NetCards.rate, Routers.rate)
- (NetCards.standard, Routers.standard)

Sophisticated Mediators Make Feasible Queries Hard to Predict

Feasible Queries FQ

- Equivalent CQ query rewritings using the views
- Might involve more than one views
- Order might matter

Query: **Feasible**

Get all **'P4' Computers**, together with their **NetCards**
and their compatible **'Wireless' Routers**

Query: **Infeasible**

Get all **Computers**

Problem

1. Large number of sources
2. Large number of views (web-services)
3. Mediator capabilities

Developer formulates an application query

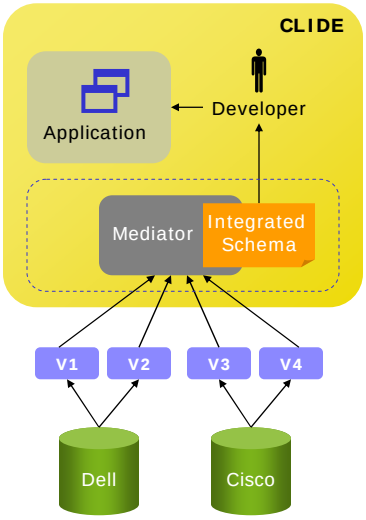
➔ Is an application query feasible?

➔ If not, how do I know which ones are feasible?

Previous options:

- The developer had to browse the view definitions and somehow formulate a feasible query
- Or formulate queries until a feasible one is found
  (trial-and-error)

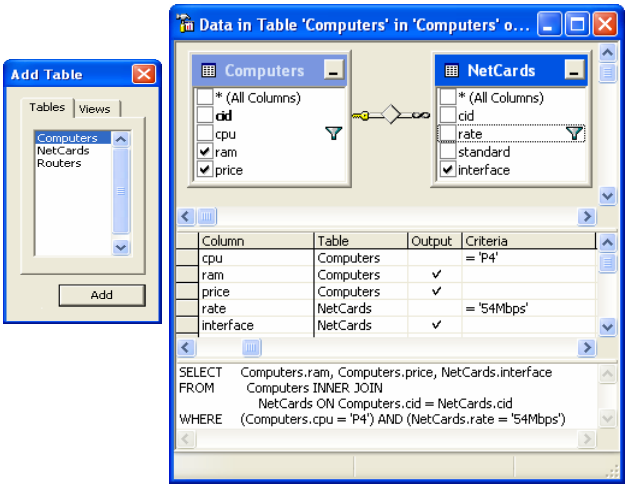
The CLIDE Solution



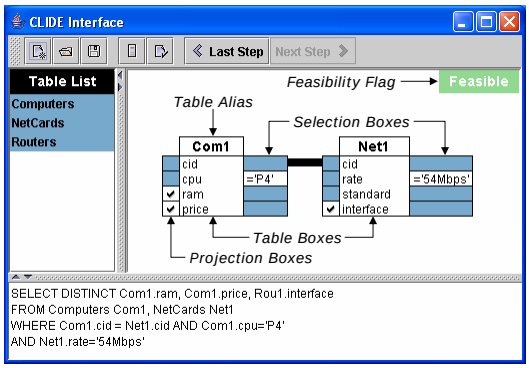
➔A query formulation interface, which interactively guides the developer toward feasible queries by employing a coloring scheme

QBE-Like Interfaces

Microsoft SQL-Server



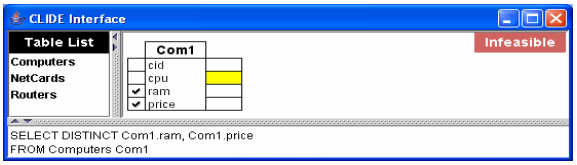
CLIDE Interface



- Table, selection, projection and join actions
- Feasibility Flag
- Color-based suggestions

Example Interaction

Snapshot 1

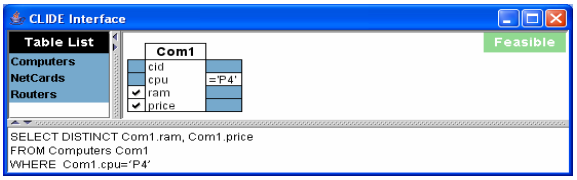


Yellow ➔ required action
- All feasible queries require this action

White ➔ optional action
- Feasible queries can be formulated w/ or w/o these actions

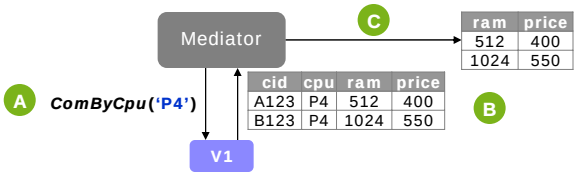
Example Interaction

Snapshot 2



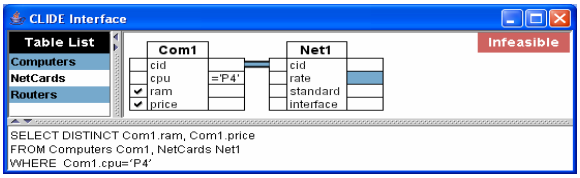
Blue ➔ required choice of action

– At least one feasible query cannot be formulated unless this action is performed



Example Interaction

Snapshot 3

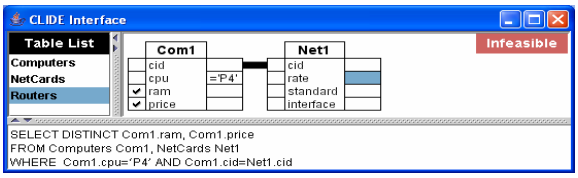


Join Lines:

- Only yellow and blue are displayed
- Must appear in Attribute Associations

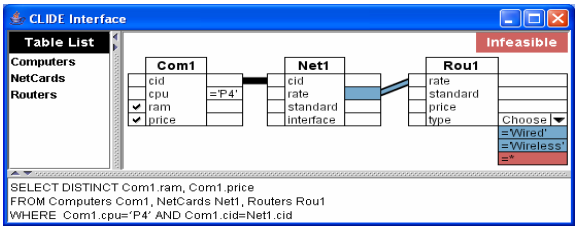
Example Interaction

Snapshot 4



Example Interaction

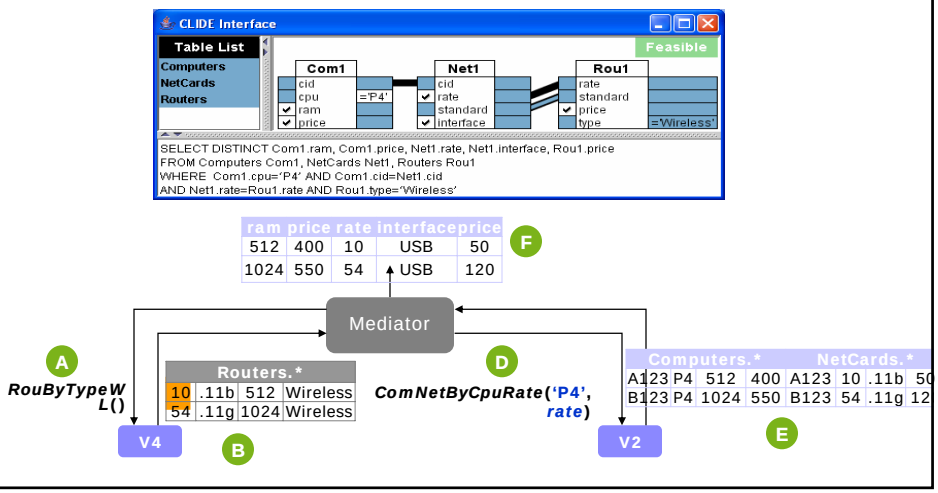
Snapshot 5



- ***** ➔ any other constant
- **Red** ➔ prohibited action
  - Does not appear in any feasible query
  - Lead to "Dead End" state

Example Interaction

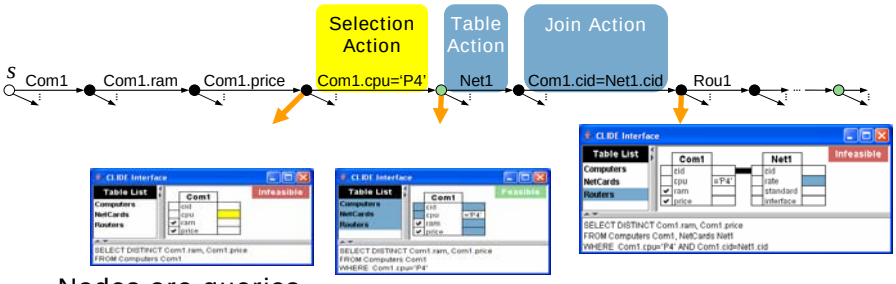
Snapshot 6



CLIDE Properties

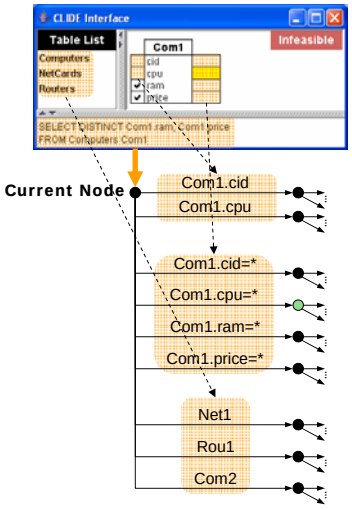
- **Rapid Convergence**
  - At every step, yellow and blue actions lead to a feasible query in a minimum number of steps
- **Completeness of Suggestions**
  - Every feasible query can be formulated by performing yellow and blue actions at every step
- **Minimality of Suggestions**
  - At every step, only a minimal number of actions are suggested, i.e., the ones that are needed to preserve completeness

Interaction Graph



- Nodes are queries
  - One for each q∈CQ
- Edges are actions
  - Table, selection, projection and join actions
- Green nodes are feasible queries
- Infinitely big structure
  - All CQ queries
  - All possible combinations of actions formulating them

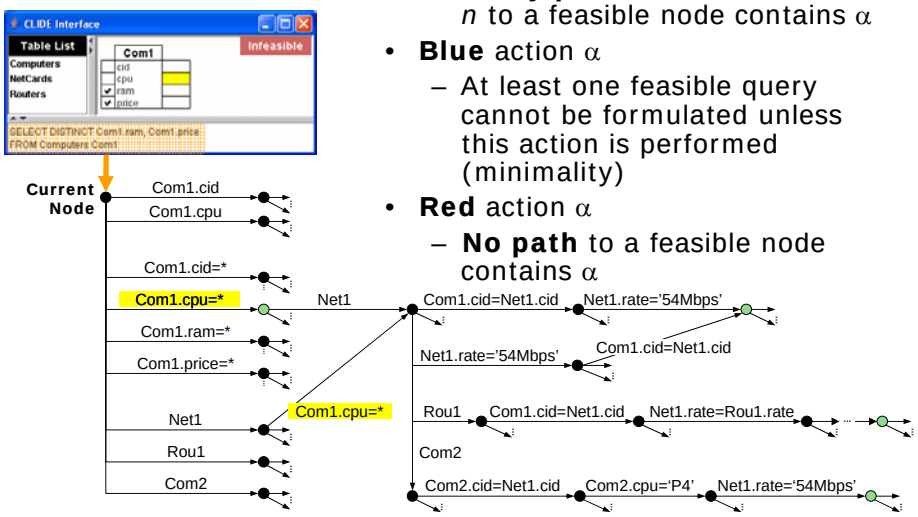
Interaction Graph: Colorable Actions



- Colorable actions $\mathbf{A_C}$ label outgoing edges of the current node

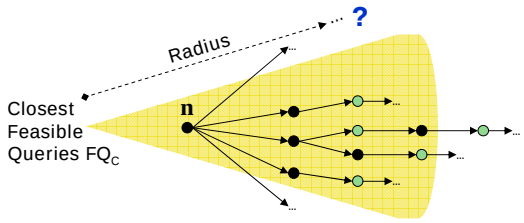
Interaction Graph: Colors

- **Yellow** action α
  - **Every path** from current node *n* to a feasible node contains α
- **Blue** action α
  - At least one feasible query cannot be formulated unless this action is performed (minimality)
- **Red** action α
  - **No path** to a feasible node contains α



Color Determined By a Finite Set of Feasible Queries

Challenge: Infinitely Many Feasible Queries

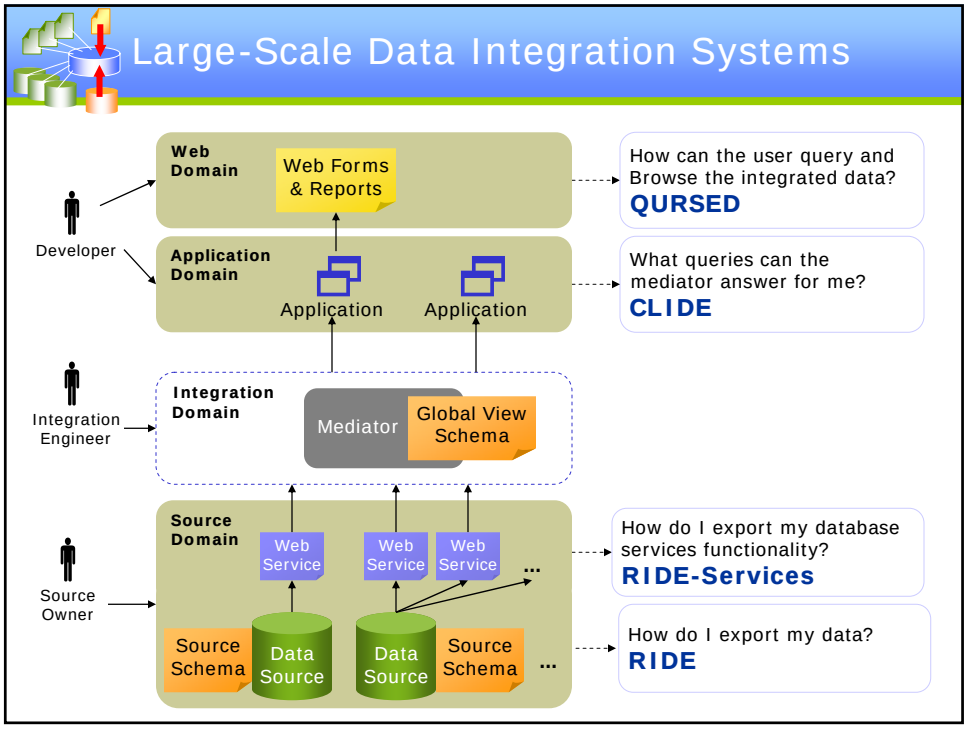


Solution: Closest Feasible Queries FQ_C

- FQ_C is sufficient to color actions in $\mathbf{A_C}$
- **Theorem:** Set of Closest Feasible Queries is Finite

Challenge: How far can the Closest Feasible Queries FQ_C be?

Solution: Based on Maximally Contained Queries FQ_{MC}


Large-Scale Data Integration Systems
Web Domain
Web Forms & Reports
Developer
Application Domain
Application
Application
Integration Engineer
Integration Domain
Mediator
Global View Schema
Source Domain
Web Service
Web Service
Web Service
...
Source Owner
Source Schema
Data Source
Data Source
Source Schema
...
How can the user query and Browse the integrated data?
QURSED
What queries can the mediator answer for me?
CLIDE
How do I export my database services functionality?
RIDE-Services
How do I export my data?
RIDE