PADAWAN

a Proxy for All Devices Accessing the World And Neighbourhood

Tree Graph Views for a Distributed Pervasive Environment

Tuyêt Trâm DANG NGOC (dntt@u-cergy.fr)
Nicolas TRAVERS (Nicolas.Travers@prism.uvsq.fr)

ETIS Laboratory, University of Cergy-Pontoise

ANR

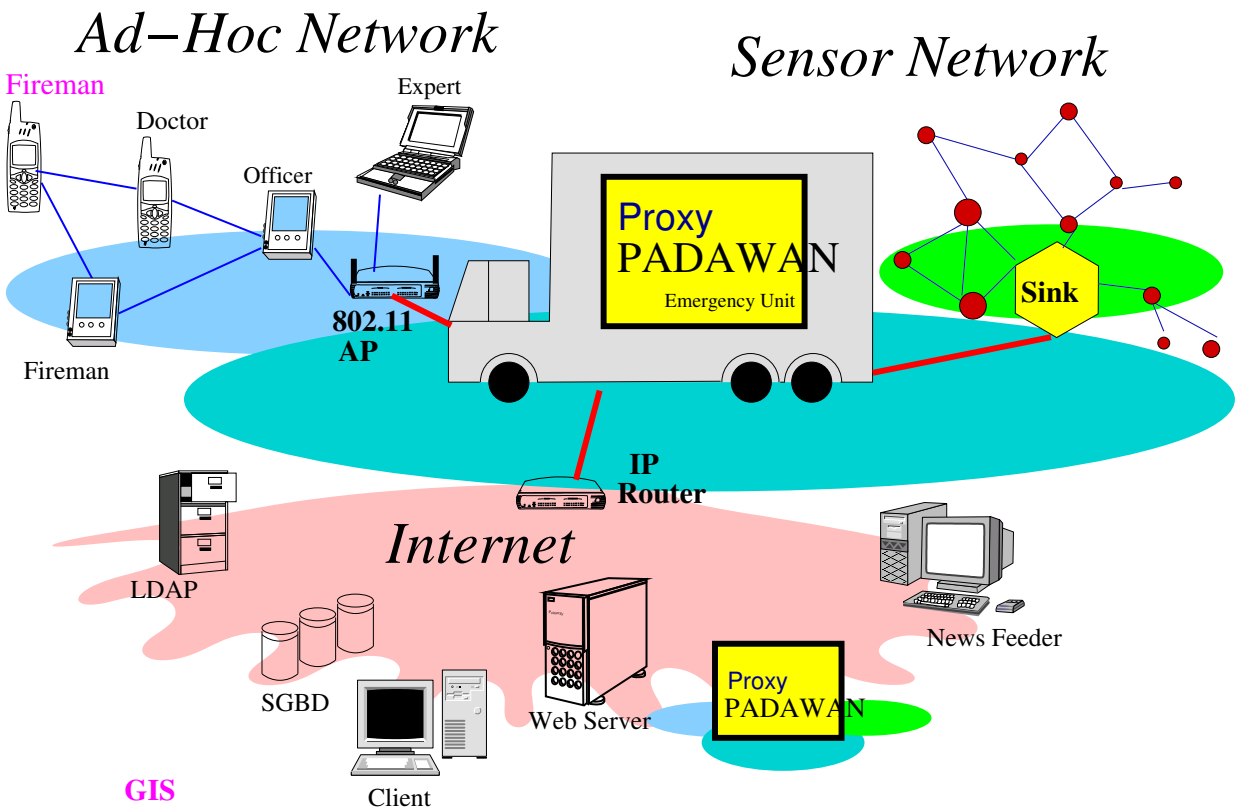
ETIS Equipe Traitement des Images et du Signal

UNIVERSITÉ de Cergy-Pontoise

Outline

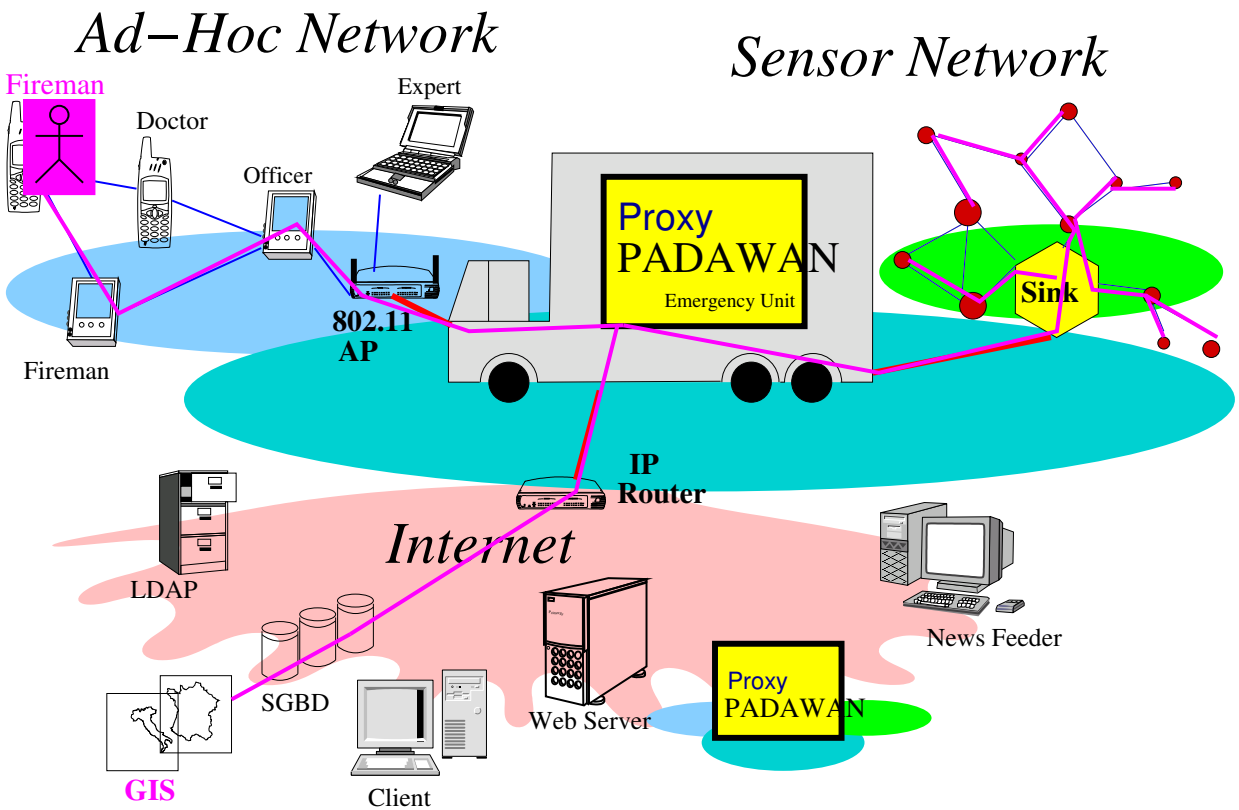
Context

Integrate the pervasive vision of computing devices in order to allow any user anywhere in the world to query anything from anywhere in the world.



Context

Integrate the pervasive vision of computing devices in order to allow any user anywhere in the world to query anything from anywhere in the world.

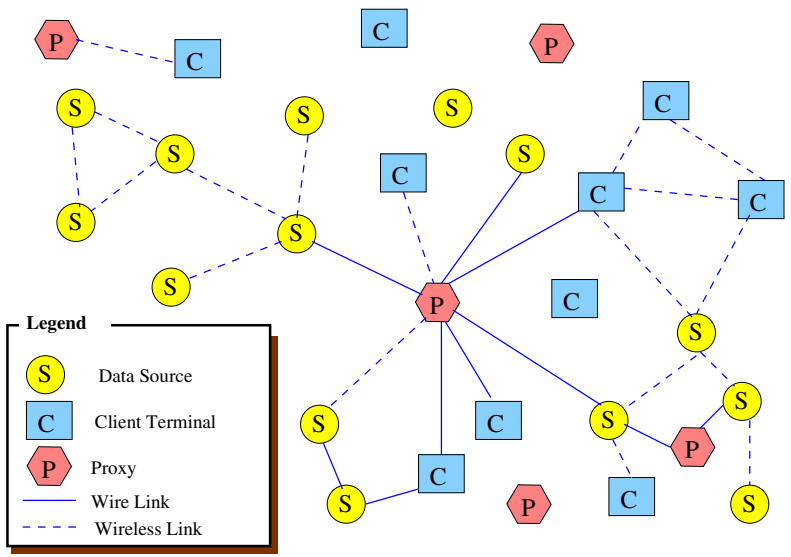


Context

How to organize the graph, so a user with any type of profile can connect on any node of the graph and access to other data that are dissiminated on nodes of the graph.

Nodes

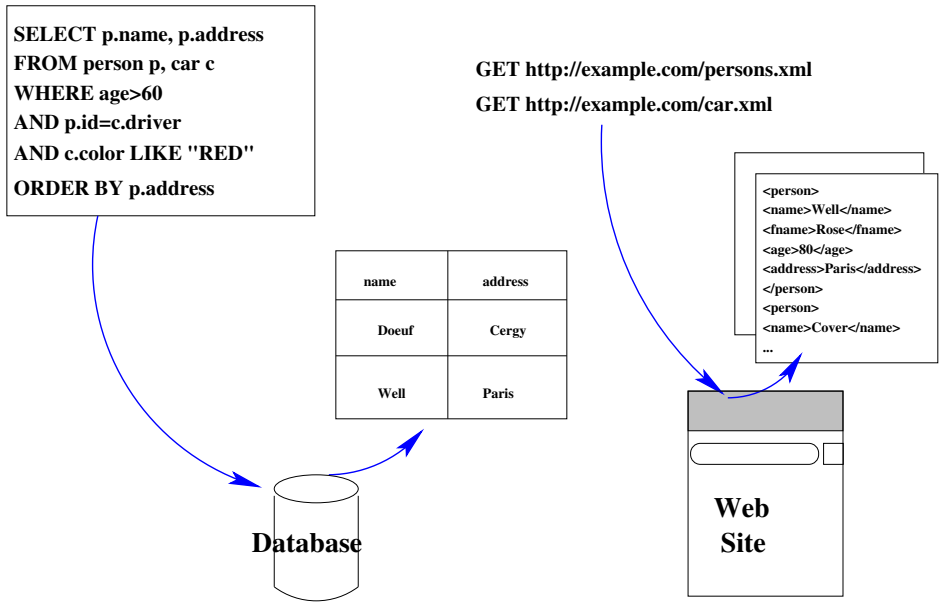
- Heterogeneous autonomous datasources
- Proxies (both client and source)
- Heterogeneous client applications



Heterogeneous Network Links

Data Integration Issues

Source Heterogeneous Data (relational, semi-structured)
→common query language and model : XML/XQuery

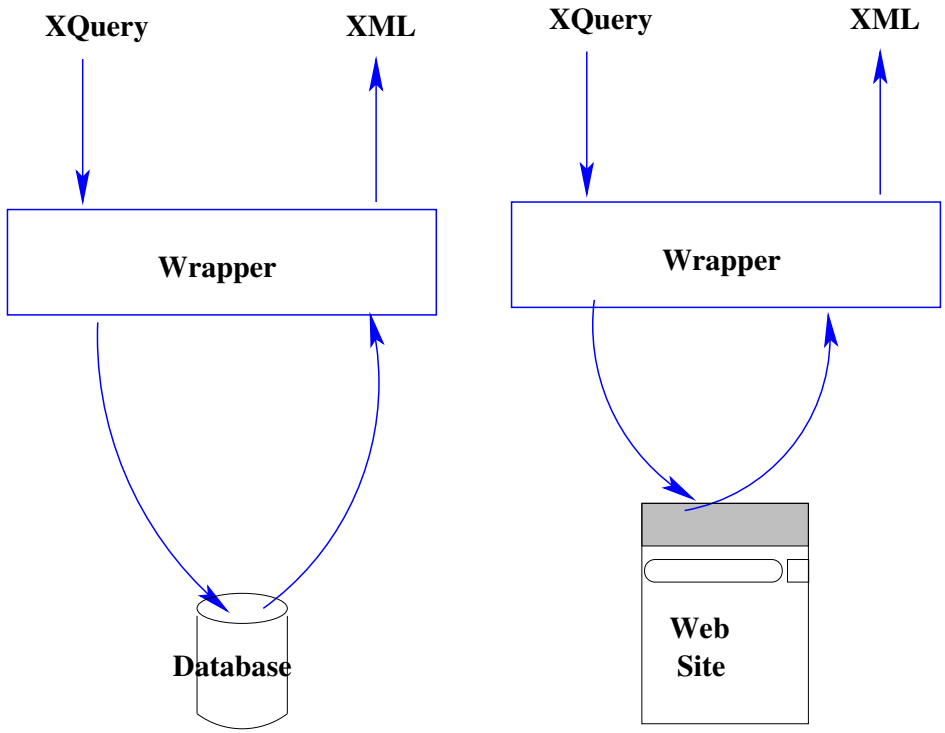


Data Integration Issues

Source

Heterogeneous Data (relational, semi-structured)

$\longrightarrow$common query language and model : XML/XQuery + Wrapper

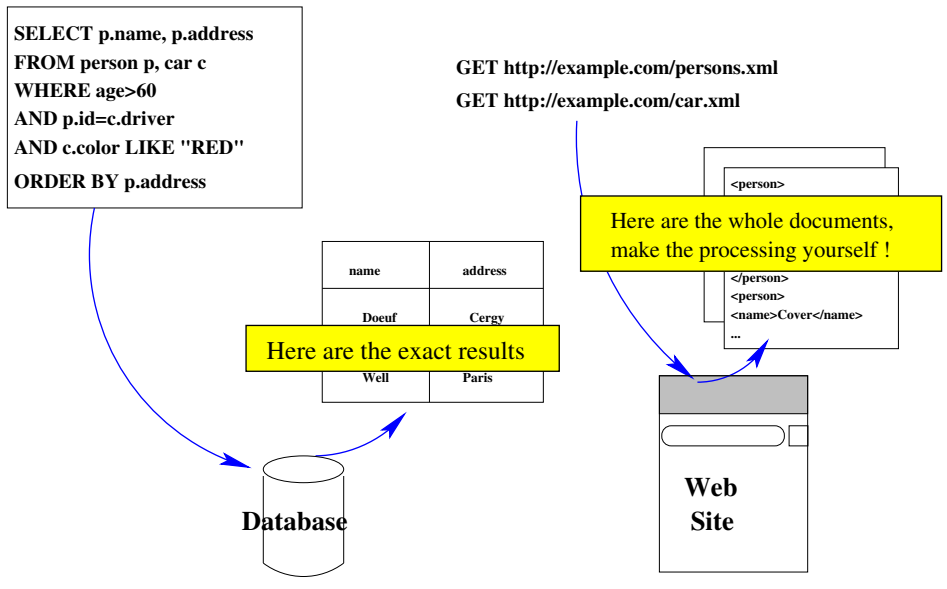


Data Integration Issues

Source

Autonomous Sources

queries more or less supported



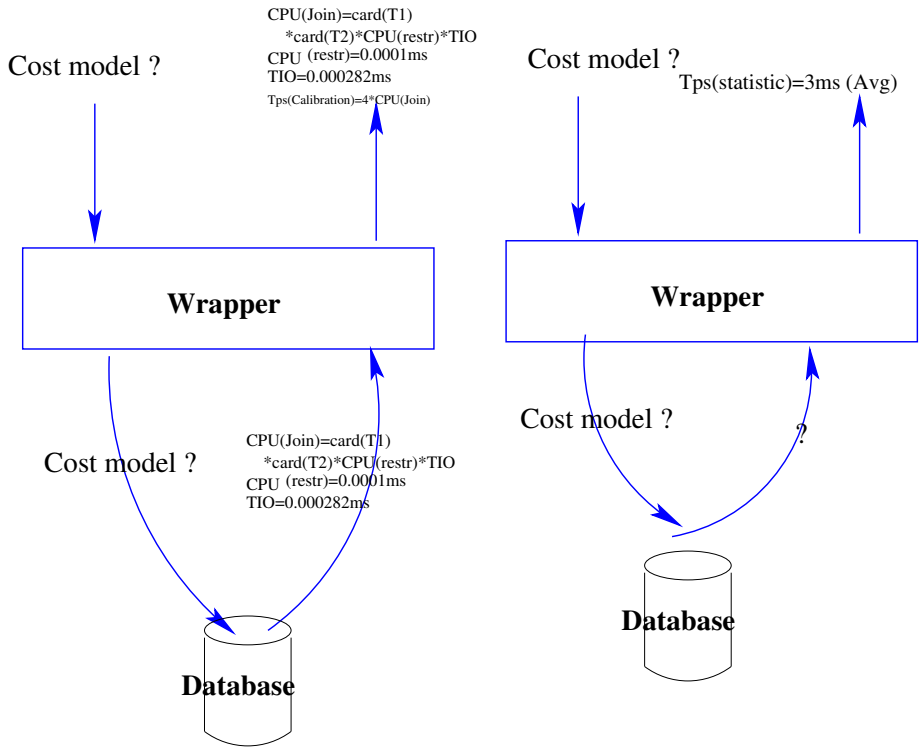
Data Integration Issues

Source

Autonomous Sources

queries more or less supported , internal information not available (cost model, statistics, schema)

→Source Description

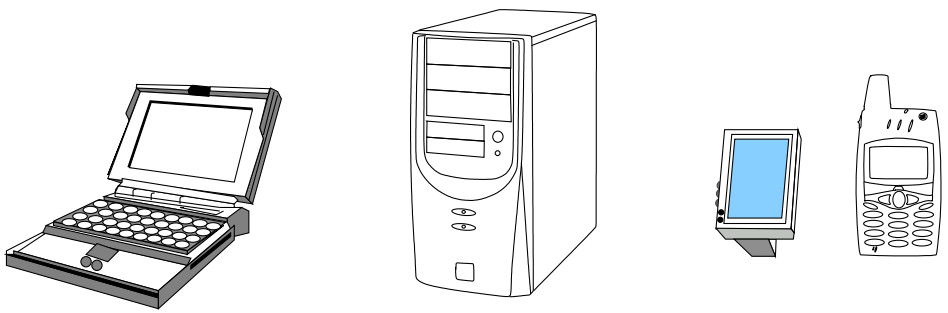


Data Integration Issues

Client Heterogeneity

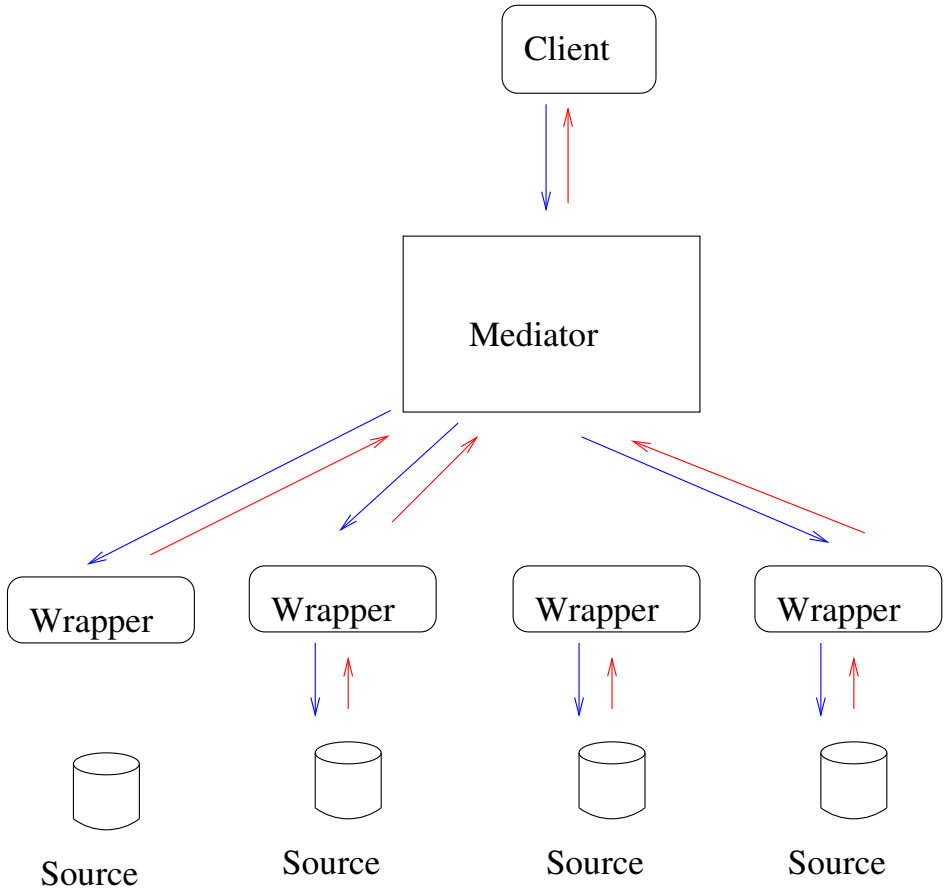
access permission, user preferences, terminal capabilities

$\longrightarrow$Client profile + View



Data Integration Issues

Data integration
evaluate query on distributed node
$\rightarrow$mediation + Source description



Data Integration Issues

- Source
  - Heterogeneous Data (relational, semi-structured)
    →common query language and model : XML/XQuery + Wrapper
  - Autonomous Sources queries more or less supported, internal information not available (cost model, statistics, schema)
    →Source Description
- Client Heterogeneity access permission, user preferences, terminal capabilities
  →Client profile + View
- Data integration evaluate query on distributed node
  →mediation + Source description

TGV : Tree Graph View [Travers and Dang-Ngoc 2004-2007]

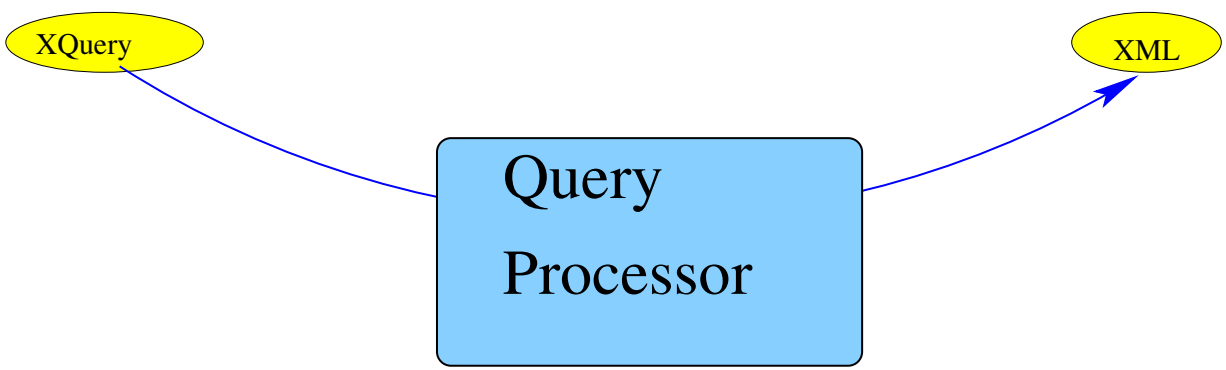
```
declare function local:f($doc as xs:string) as element()
{for $x in (doc(?rev.xml?)/review|doc(?$doc?)/catalog)
[. contains(?Robin Hobb?)]/book/[.//price > 15]
where
some $y in $x/comments
satisfies contains ($y, ?Excellent?)
order by $x/@isbn
return
<book>
{$x/@isbn} <price>{$x//price/text()}</price>
{ if (count($x/title) > 2)
then
{ for $z in doc(?books.xml?)/book
where $z/@isbn = $x/@isbn
return <title>{($z/title)[3]}</title>
}else<title/>
} </book>
}
```

TGV : Tree Graph View [Travers and Dang-Ngoc 2004-2007]

XQuery has a (too) rich syntax :

- XPath
- Constraints, Filter
- Quantifiers
- Document construction
- Nesting, Aggregates, Sorts
- Conditional operators, Set operators
- Sequences, Function

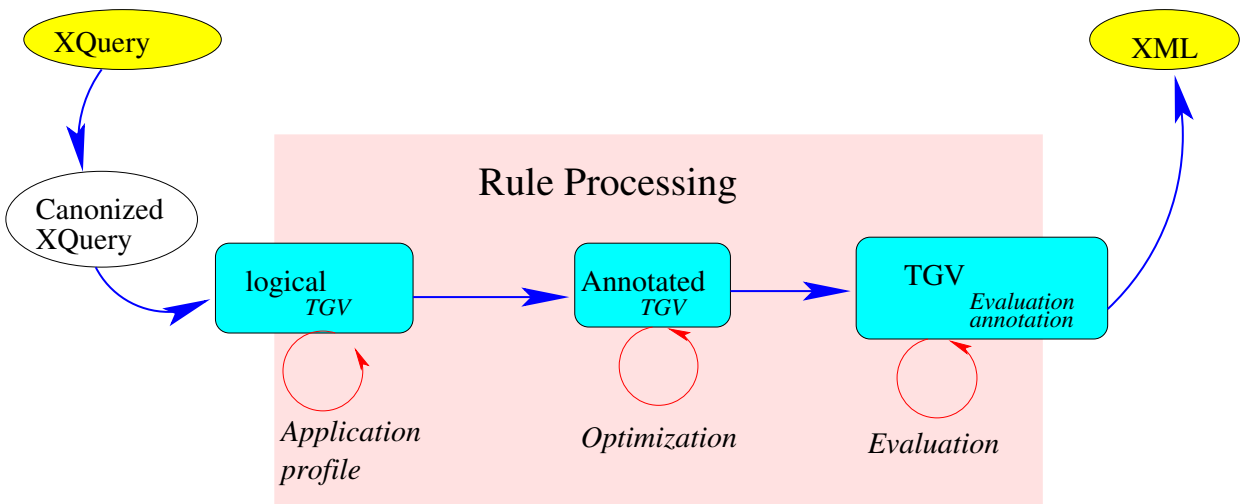
Hard to parse, process and identify subpart and dependancies.



TGV : Tree Graph View [Travers and Dang-Ngoc 2004-2007]

Model based on Tree Pattern matching that :

- Supports Full untyped XQuery [WEBIST2007]
- Provides an intuitive representation [DASFAA 2007]
- Designed for distributed environment [IBIS Journal 2006]
- Support for optimisation and for evaluation [ICEIS 2007]



TGV : Example

"list every buildings occupied by more than 100 inhabitants, and for each, get the district and the list of maximum temperature measured by the sensors located in the same district."

```
for $a in /buildings/building
where $a/description/inhabitant > 100
return
  <districtMonitoring>
    <location> {$a/district} </location>
    <temperatures>
        { for $b in //sensor
        where
          $b/deploymentArea/district = $a/district
        return
          <temperature>
            {$b/max_temp}
          </temperature>}
    </temperatures>
  </districtMonitoring>
```

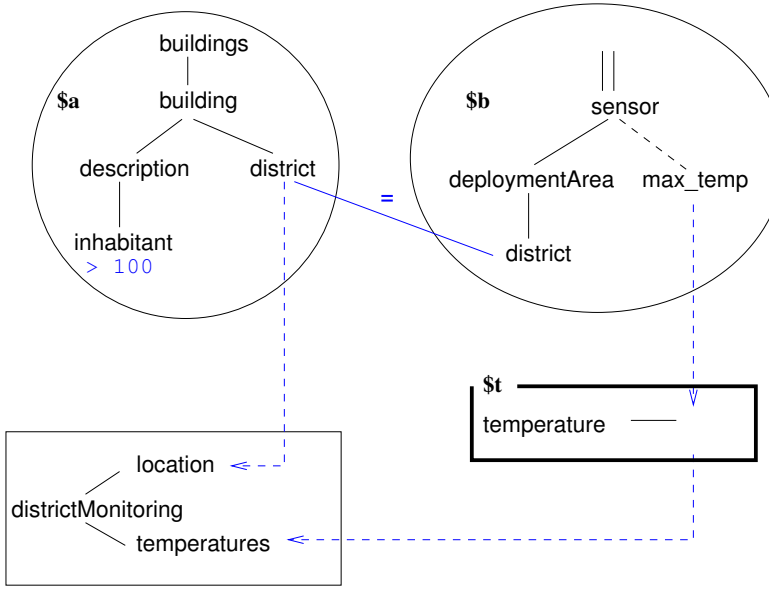
```
<districtMonitoring>
  <location> Yellow Lake </location>
  <temperatures>
    <temperature> 14 </temperature>
  </temperatures>
</districtMonitoring>
<districtMonitoring>
  <location> Green Valley </location>
  <temperatures>
    <temperature> 163 </temperature>
    <temperature> 25 </temperature>
    <temperature> 43 </temperature>
  </temperatures>

</districtMonitoring>
```

TGV : Example

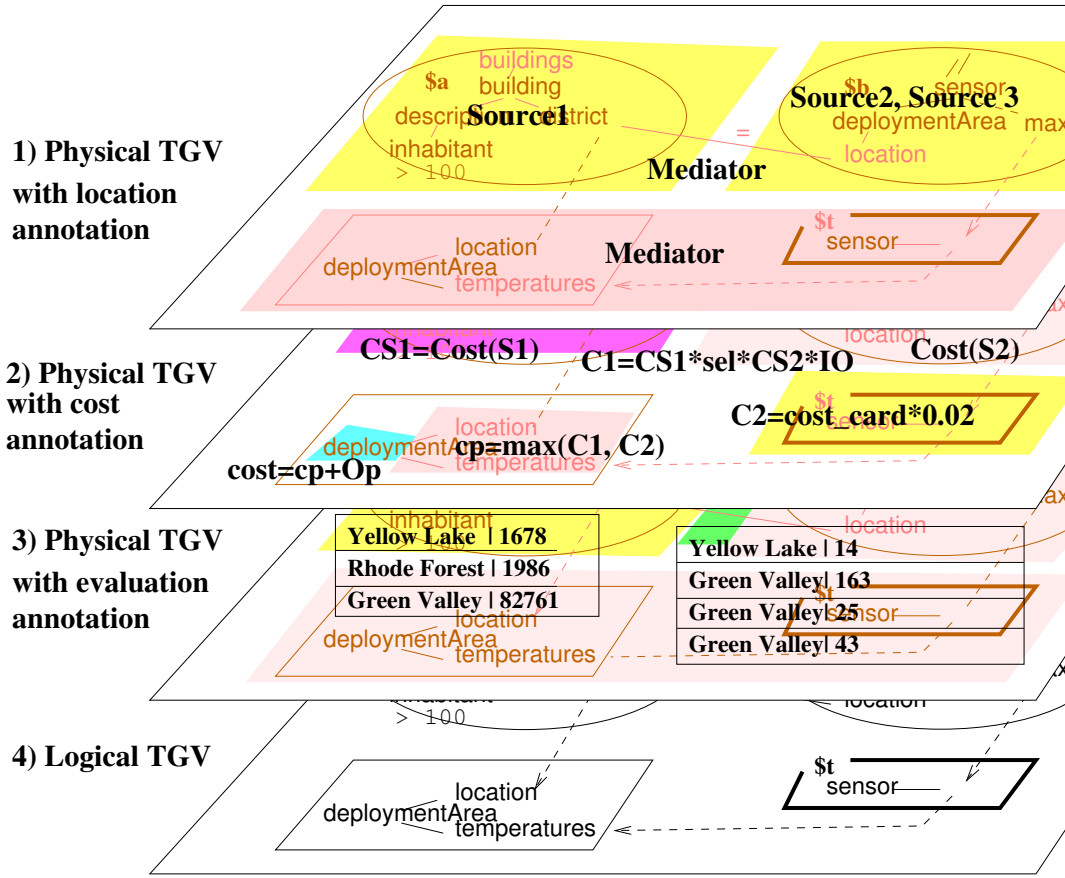
"list every buildings occupied by more than 100 inhabitants, and for each, get the district and the list of maximum temperature measured by the sensors located in the same district."

```
for $a in /buildings/building
where $a/description/inhabitant > 100
return
  <districtMonitoring>
    <location> {$a/district} </location>
    <temperatures>
        { for $b in //sensor
        where
          $b/deploymentArea/district = $a/district
        return
          <temperature>
            {$b/max_temp}
          </temperature>}
    </temperatures>
  </districtMonitoring>
```



TGV : Generic Annotation for any type of Information

- A layer per information type
- Information on any set of TGV elements (including annotation)
- Any granularity, possible recovering
- Generic Annotation (ex. several type of cost model with complex formulas)



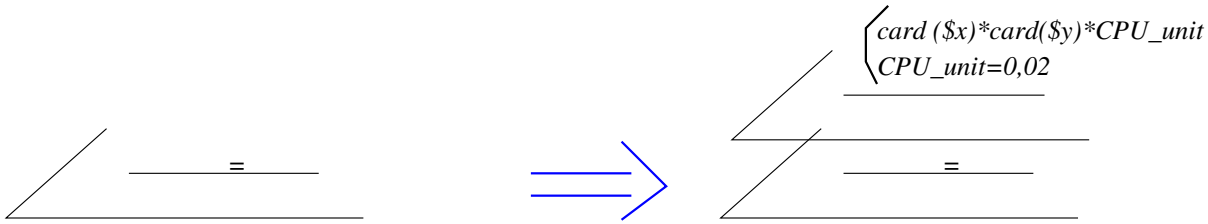
TGV : Rules

Rule

Annotated TGV Pattern condition" $\longrightarrow$ Annotated TGV Pattern Conclusion"

Rules for :

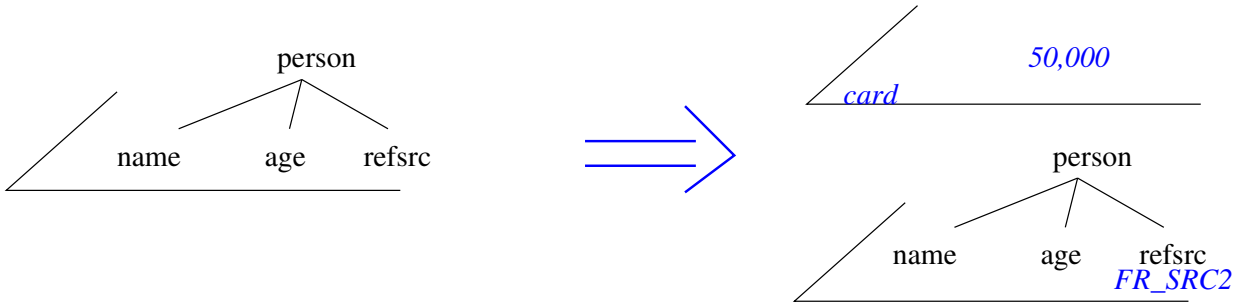
TGV : Rules



Rules for :

- Local Transformation

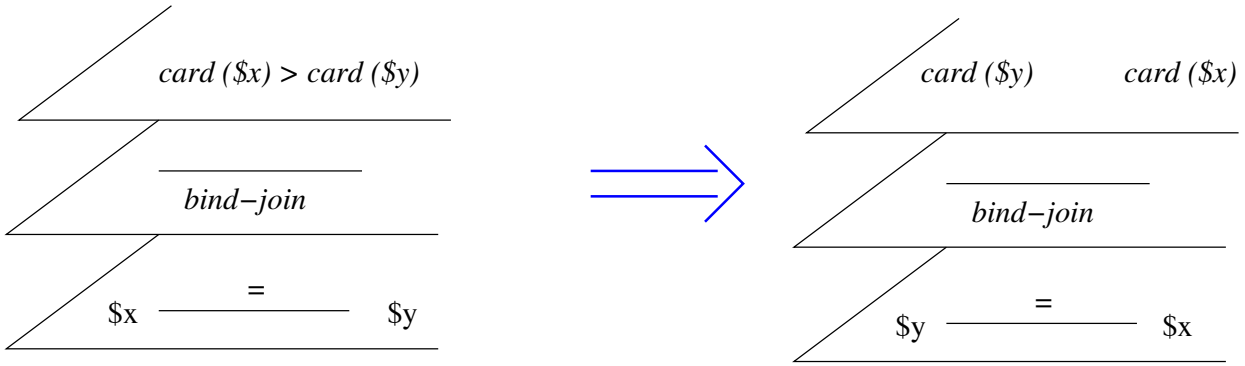
TGV : Rules



Rules for :

- Local Transformation

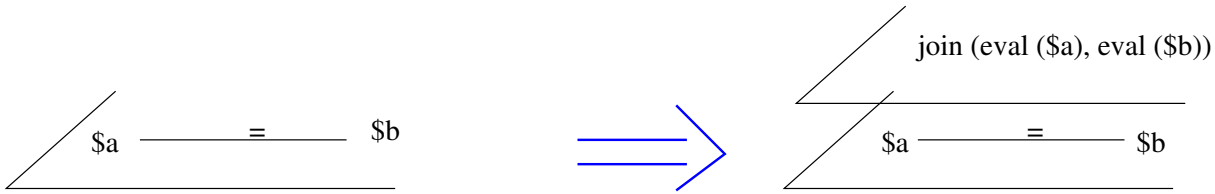
TGV : Rules



Rules for :

- Local Transformation
- Optimization

TGV : Rules



Rules for :

- Local Transformation
- Optimization
- Local Evaluation

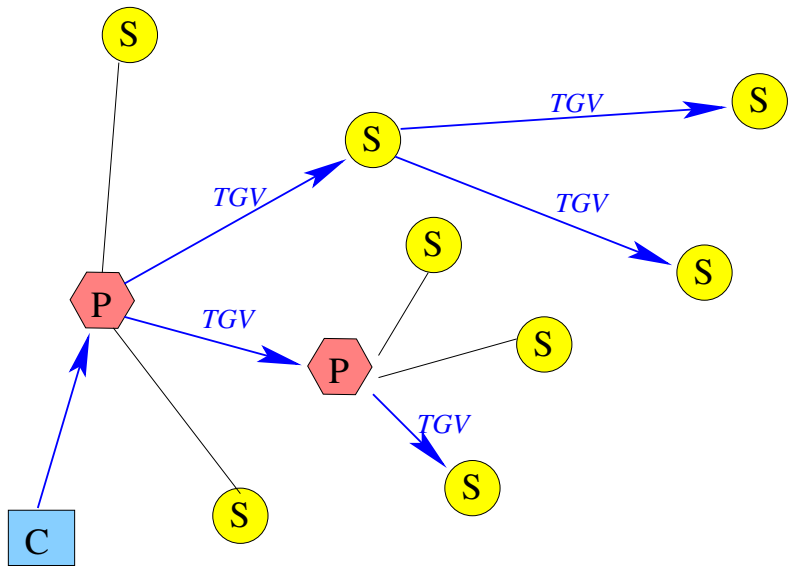
TGV : Rules

Rules for :

- Local Transformation
- Optimization
- Local Evaluation
- Distributed Evaluation

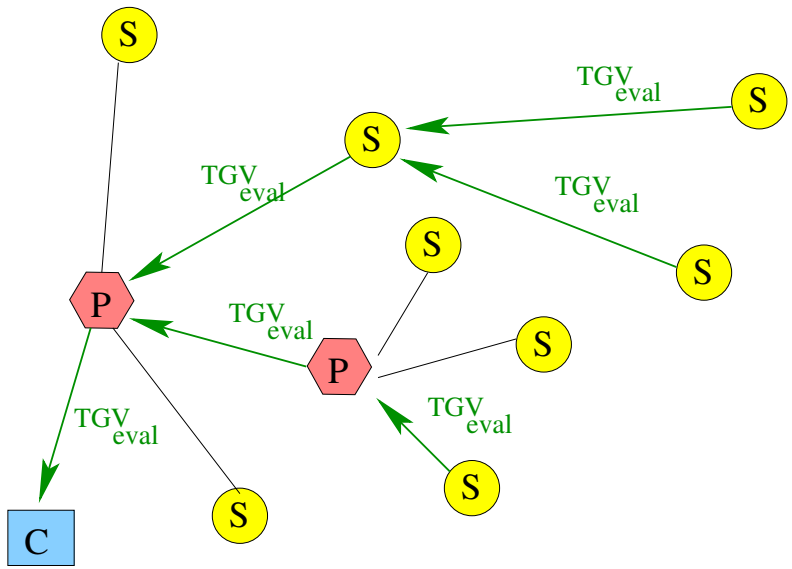
TGV Mobile Agent

- Initiated by the client
- Evaluated on the fly
- Parallel evaluation :
  - cloning the TGV on some nodes.
  - evaluation rules located on the node.
- Leaves of the recursion tree on source node. Source node fill the evaluation annotation layer.



TGV Mobile Agent

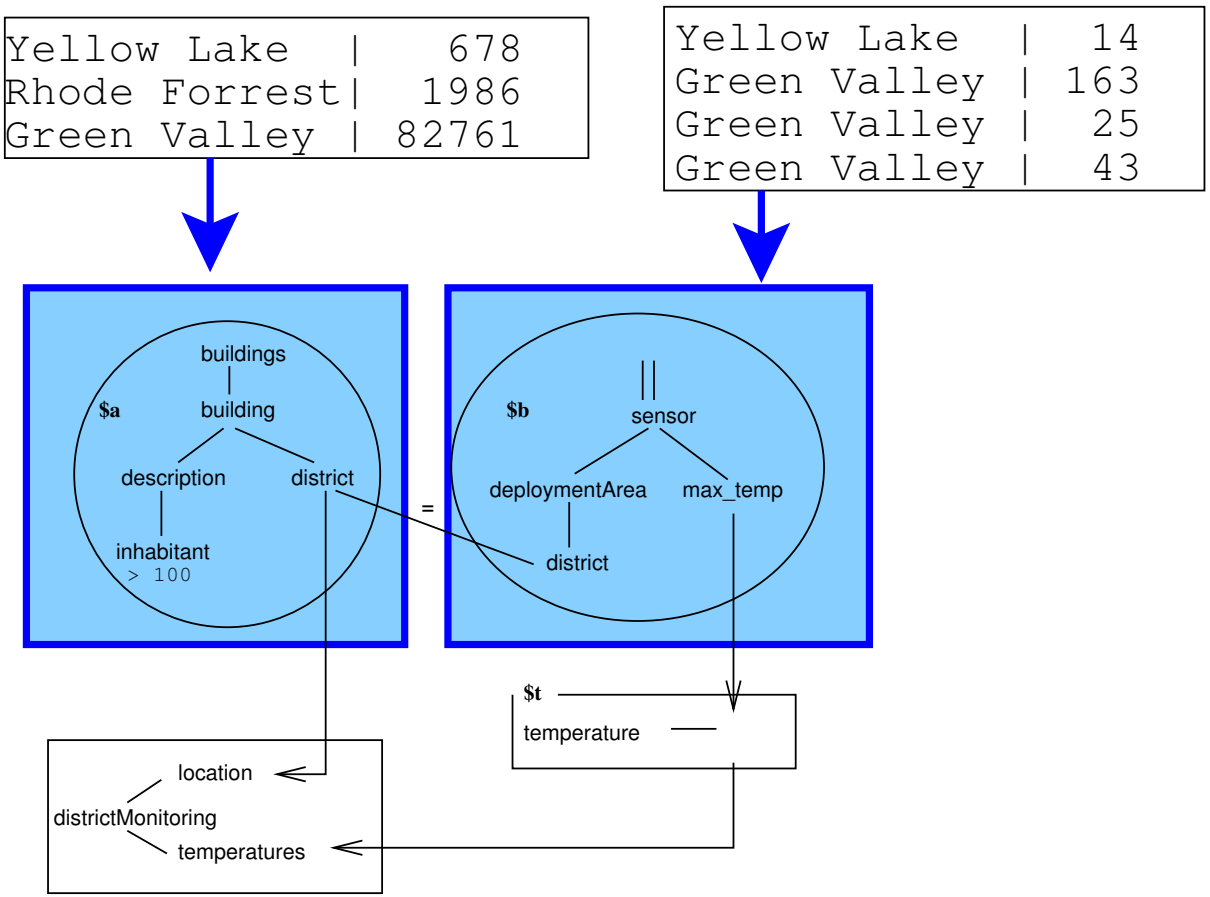
- Initiated by the client
- Evaluated on the fly
- Parallel evaluation :
  - cloning the TGV on some nodes.
  - evaluation rules located on the node.
- Leaves of the recursion tree on source node. Source node fill the evaluation annotation layer.



TGV : Evaluation

Evaluation rules :

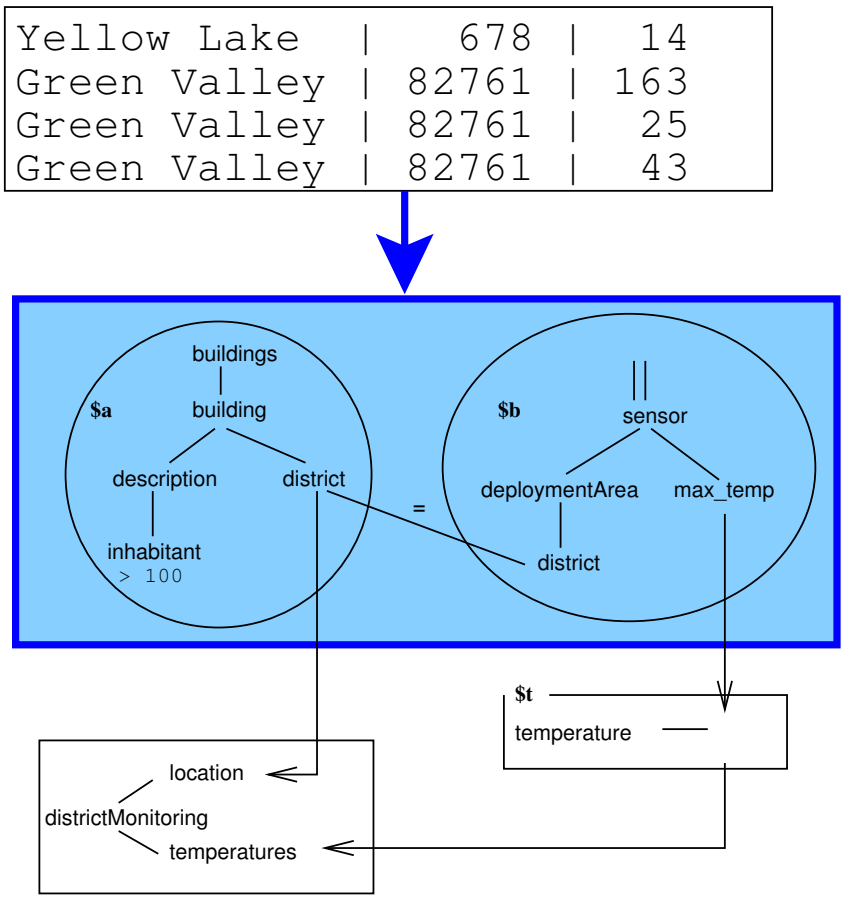
- Evaluation on Source Wrappers



TGV : Evaluation

Evaluation rules :

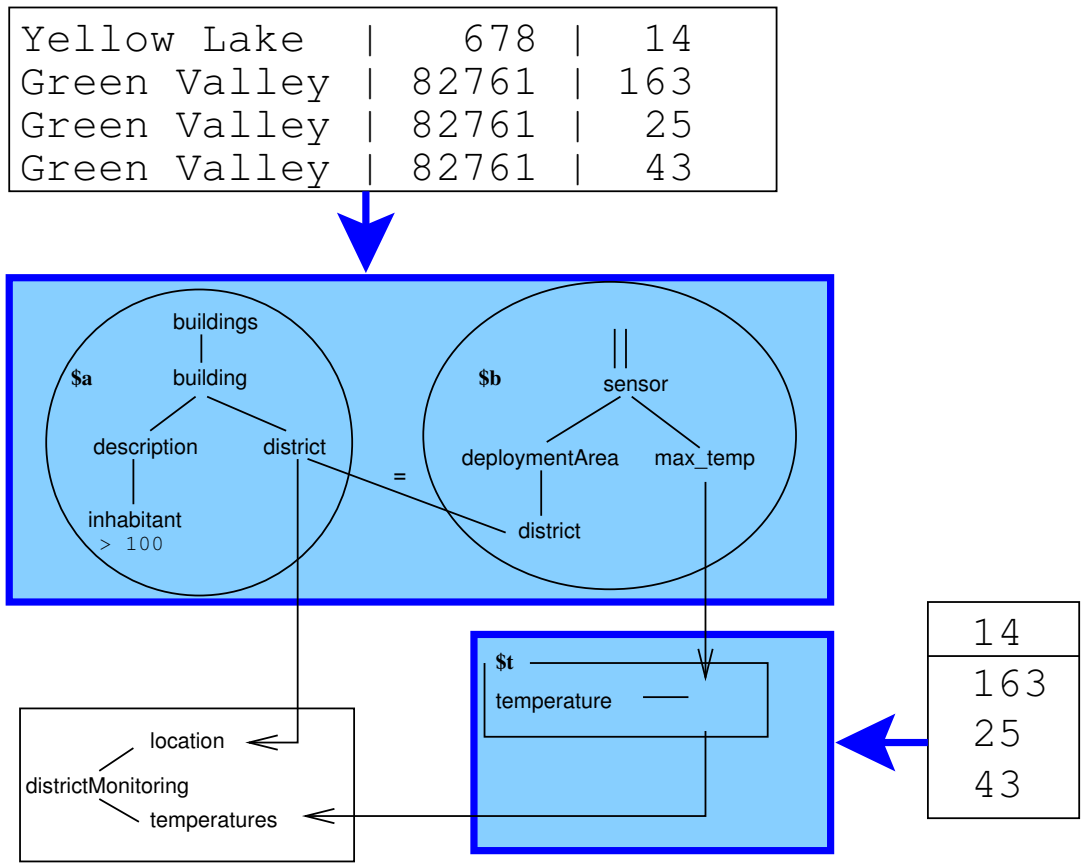
- Evaluation on Source Wrappers
- Join



TGV : Evaluation

Evaluation rules :

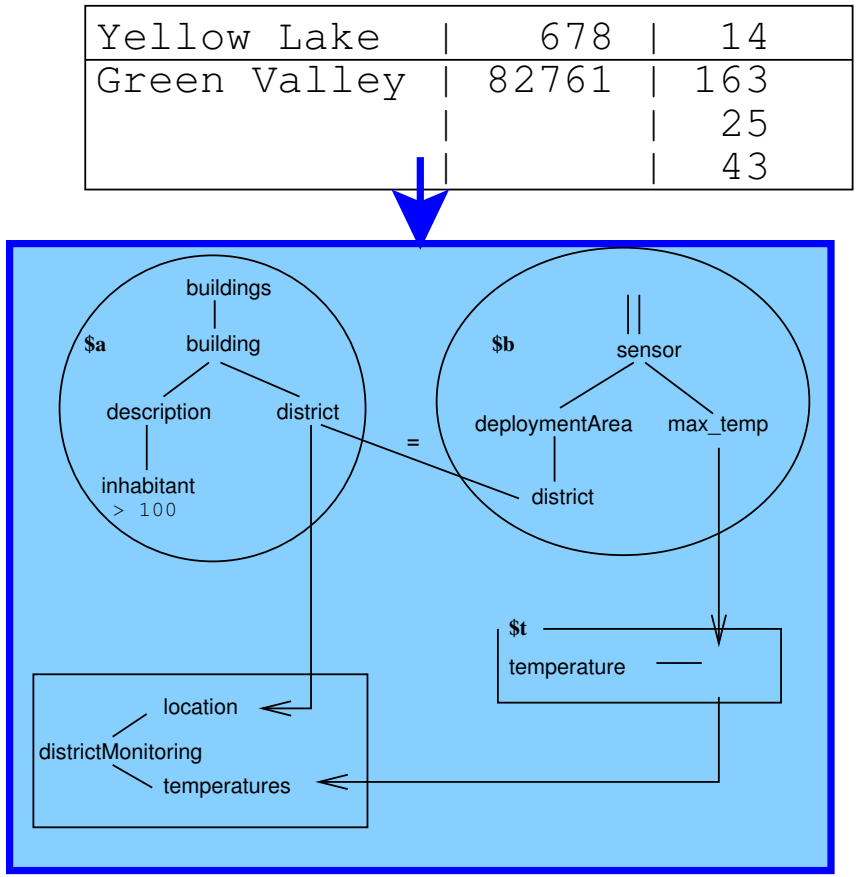
- Evaluation on Source Wrappers
- Join
- Nest



TGV : Evaluation

Evaluation rules :

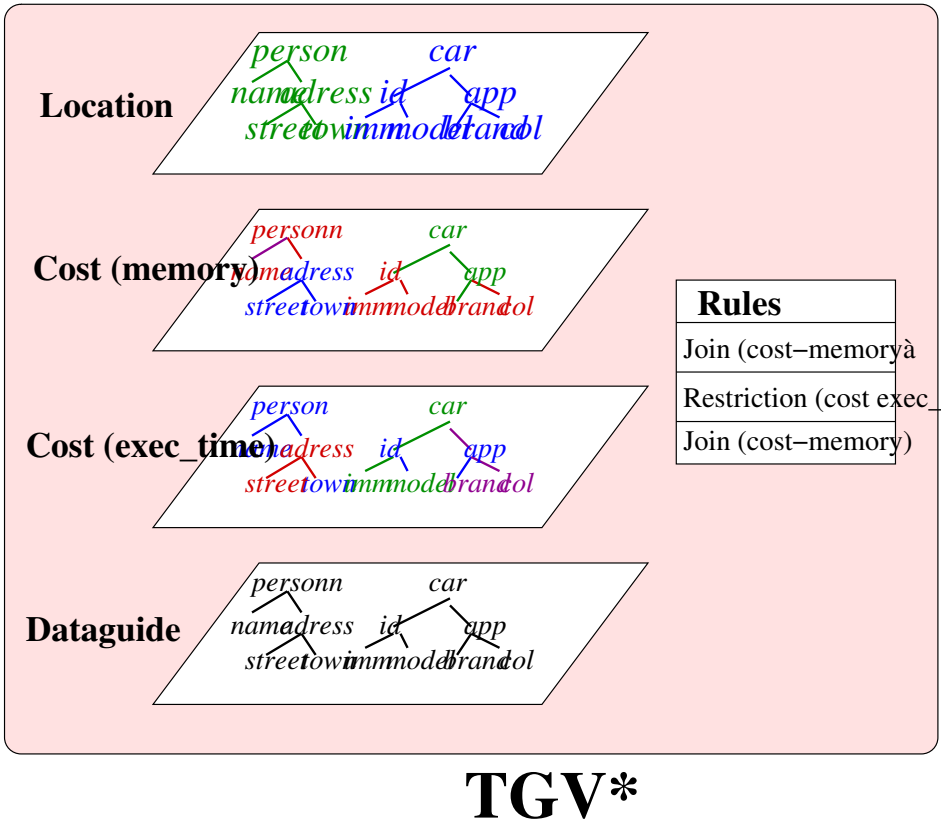
- Evaluation on Source Wrappers
- Join
- Nest
- Final projection



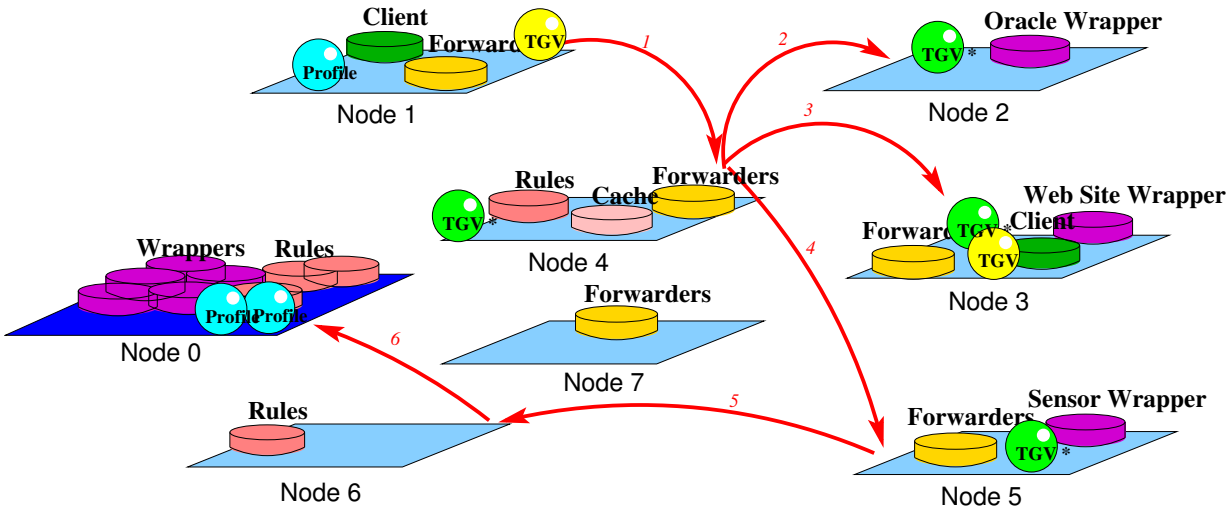
TGV* : Sources Description

How to annotate the TGV ?
$\longrightarrow$TGV* to describe sources in an exhaustive manner.

- Dataguide : source part (STP) of the TGV with annotation
- Annotations
- Set of rules (for operators)



PADAWAN : a Multi-Agent Plateform



It is the agents located on the node that define the functionalities of the node.

- Client Node : Client and Presentation Agents
- Source Node : Wrapper Agent
- Proxy Node : Cache, Router, and Rules Agents
- Repository Node : Agent Library

Prototype PADAWAN (a Plateform for All Devices Accessing the World And Neighbourhood) on http ://padawan.ensea.fr/

Conclusion

TGV Mobile Agent

- Full untyped-XQuery evaluation
- Evaluation through the Multi-Agents Platform PADAWAN graph
- Dynamic Optimization

Current Work : TGV* Mobile Agent

- describe data
- generic enough to describe all information known on a source.
- abstract TGV*

Publications on TGV and PADAWAN

Dang-Ngoc, T.-T. and Travers, N. (2007).
Tree graph views for a distributed pervasive environment.
In *the 1st International Conference on Network-Based Information Systems (NBIS)*, Regensburg, Germany.

Liu, T., Dang-Ngoc, T.-T., and Laurent, D. (2007).
Cost framework for a distributed semi-structured environment.
In *in the proceedings of the International workshop Database Management and Application over Networks - DBMAN (APWeb/WAIM Workshop)*, Huangshan, China.

Travers, N. and Dang-Ngoc, T.-T. (2007).
An extensible rule transformation model for xquery optimization.
In *The 9th International Conference on Enterprise Information Systems (ICEIS)*, Madeira, Portugal.

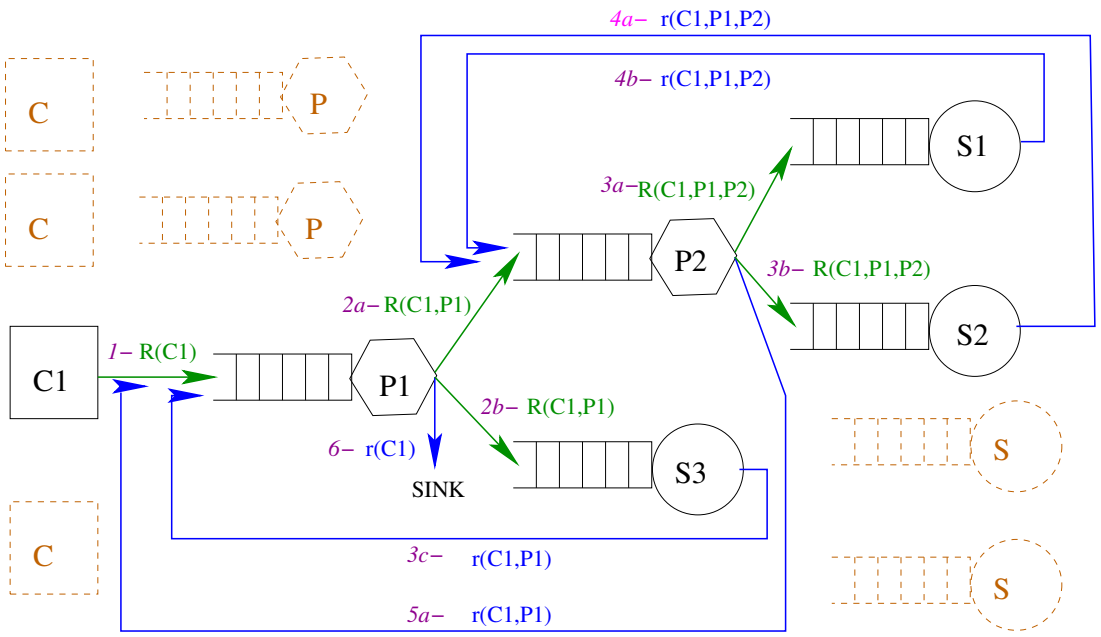
Travers, N., Dang-Ngoc, T.-T., and Liu, T. (2006).
Tgv : an efficient model for xquery evaluation within an interoperable system.
International Journal of Interoperability in Business Information Systems (IBIS), 3.
ISSN : 1862-6378.

Travers, N., Dang-Ngoc, T.-T., and Liu, T. (2007a).
An efficient evaluation of xquery with tgv.
In *the 3rd International Conference of WEB Information Systems and Technologies (Web-IST)*, Spain.

Travers, N., Dang-Ngoc, T.-T., and Liu, T. (2007b).
Full untyped xquery canonisation.
In *in the proceedings of the International workshop on Emerging Trends of Web Technologies and Applications -WebETrends (APWeb/WAIM Workshop)*, Huangshan, China.

Travers, N., Dang-Ngoc, T.-T., and Liu, T. (2007c).
Tgv : a tree graph view for modelling untyped xquery.
In *the 12th International Conference on Database Systems for Advanced Applications (DASFAA)*, Thailand.

Discrete Event Simulation

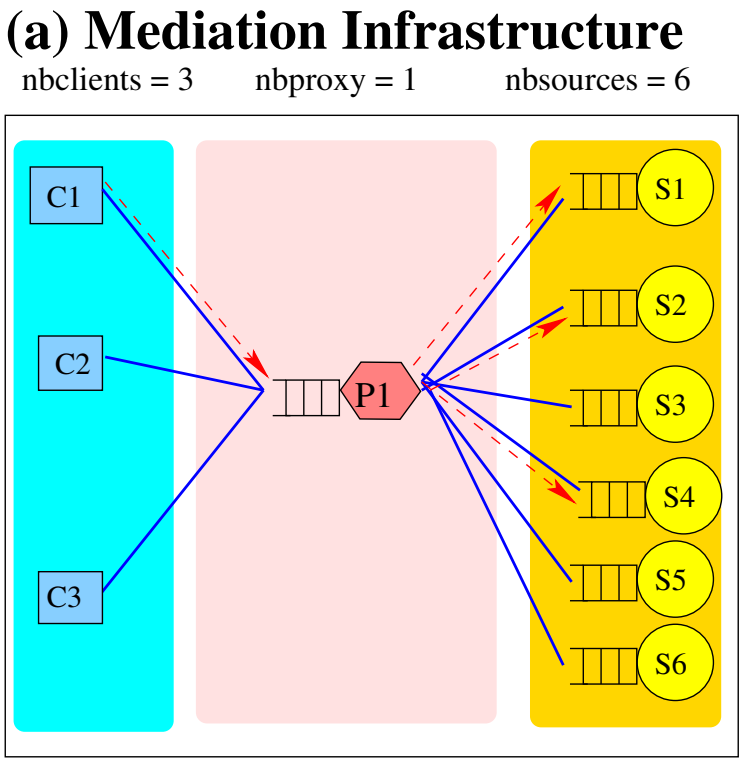


- Sources already located located
- route of the request randomly generated, using randomly 0-N sources
- time of service, and queries generation parameterized

Simulation

Mediation Architecture

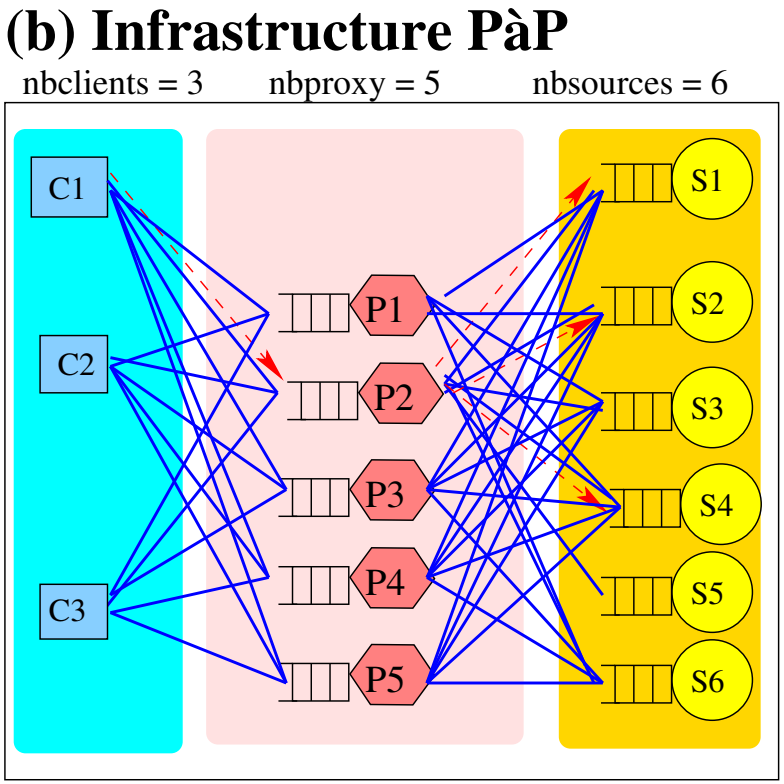
- Direct Access to sources
- Bottleneck on mediator



Simulation

P2P Network

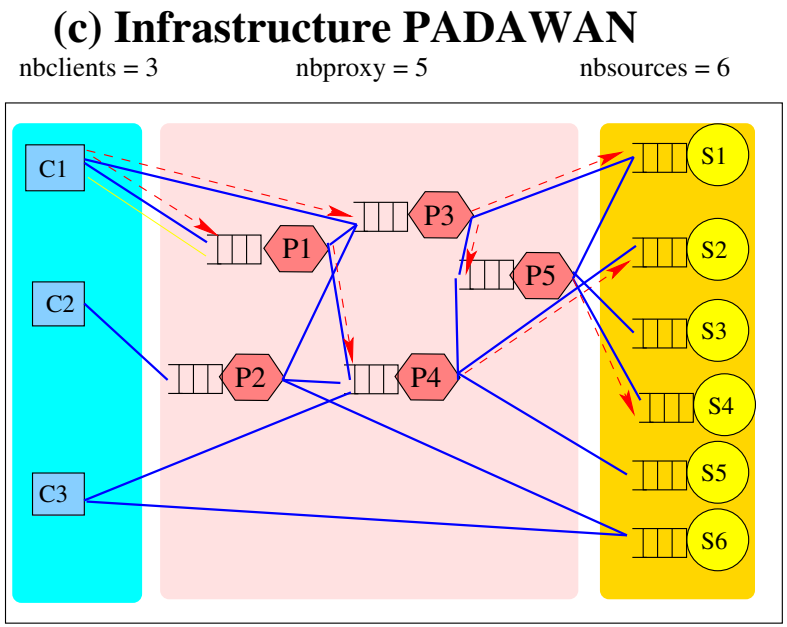
- Each Peer is a mediator
- Direct Access to sources
- Load on each peer
- Each peer knows how to resolve a whole request



Simulation

Padawan Network

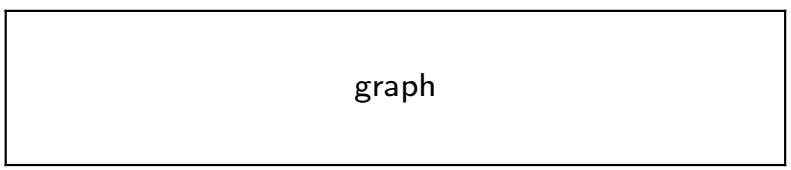
- A peer has access to some other peers and eventually to some sources
- Distribution of the query evaluation
- Evaluation depending on the peers capabilities
- Distributed waiting time



Simulation Result

Times / Number of queries to process in the system 20 queries generator (=clients), 100 sources, For P2P and PADAWAN : 20 peers.

- Mediator : Bottleneck
- P2P Architecture better thant PADAWAN Architecture
- Linear



Simulation Result

Times / Number of queries to process in the system 20 queries generator (=clients), 100 sources, For P2P and PADAWAN : 20 peers.

- Mediator : Bottleneck
- P2P Architecture better thant PADAWAN Architecture
- Linear

graph-tw

But :
Mean waiting time better in PADAWAN

Simulation limitation

- Opening connection delay
  - PADAWAN : few connection per proxy
    →all connections can be left opened.
  - P2P : each peer can potentially reach any source
    →can't maintain all opening/closing connection
    →consider connection delay
- Nodes capabilities
  - In the major P2P networks, equivalent nodes with same capabilities (or 2 or 3 categories with Superpeer and normal peer)
  - PADAWAN queries can be evaluated on nodes with different capabilities