EXTRACTING DATA FROM THE WEB

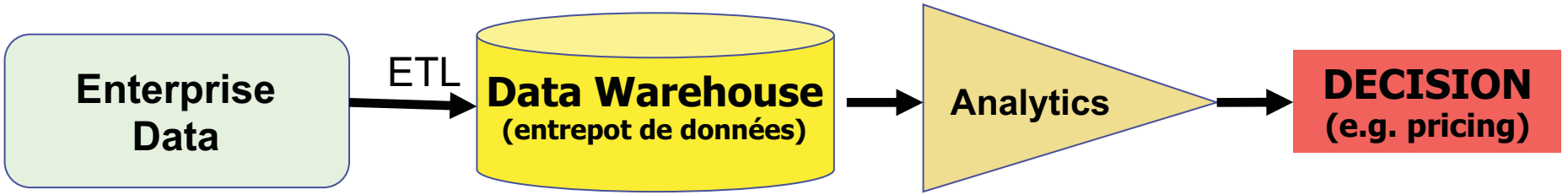
Georg Gottlob

Oxford University

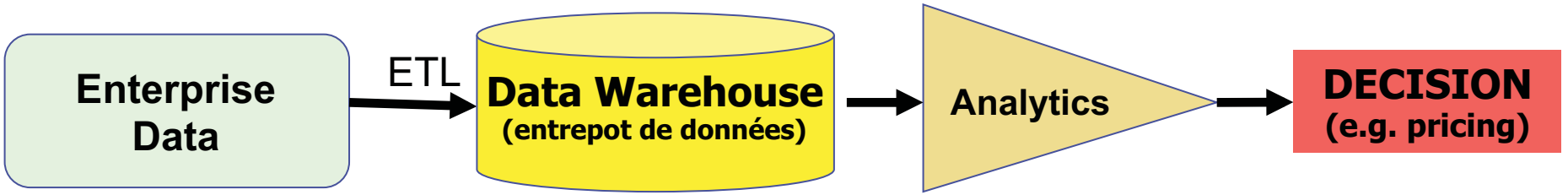
Talk Outline

- Motivation: need of information extraction
- Logical foundations of information extraction
- The Lixto Visual Wrapper
- The Diadem Project

Traditional data-based decision making in enterprises.



Traditional data-based decision making in enterprises.



But often the most relevant data are outside the company, **on the Web!**

→ Online data intelligence, online market intelligence, automatic web data extraction.

Online Market Intelligence (OMI)

(surveillance du marché)

- **Electronics Retailer (**détaillant d'électronique - composants**) :**
  market overview, 20 competitors, 200,000 products/prices

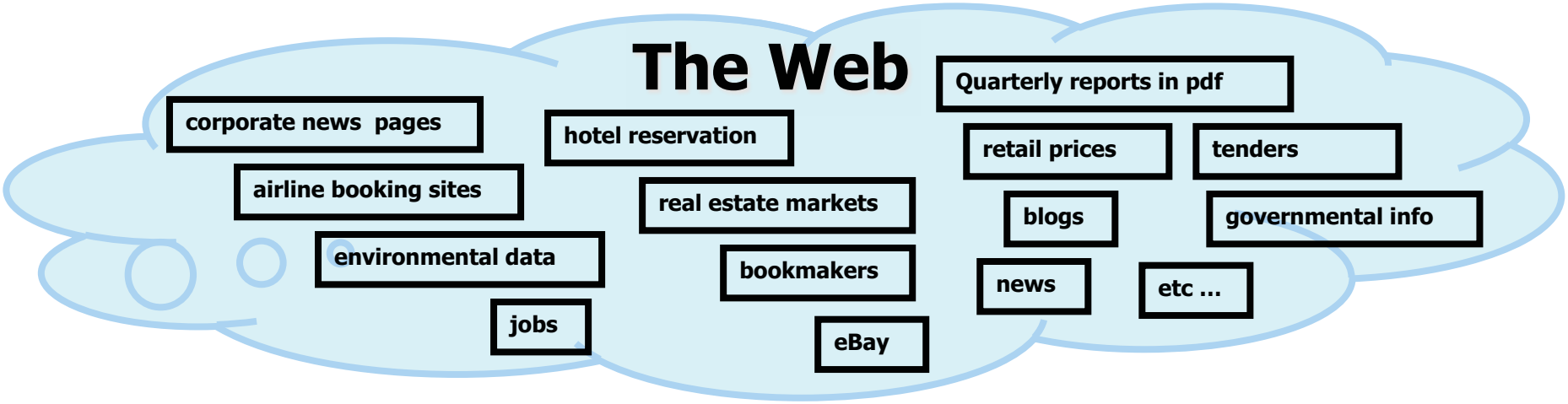
- **Supermarket Chain:** Price comparison; must quickly react to special offers (offres spéciales) , new products,…

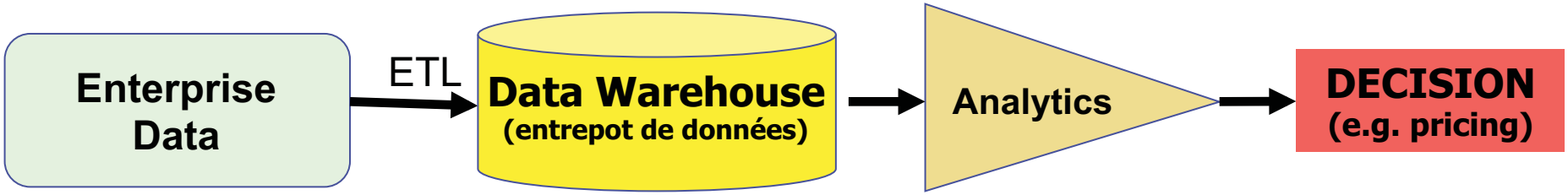
- **Internet Travel Agency:** Gives best price guarantee, wants to detect "pricing attacks",…

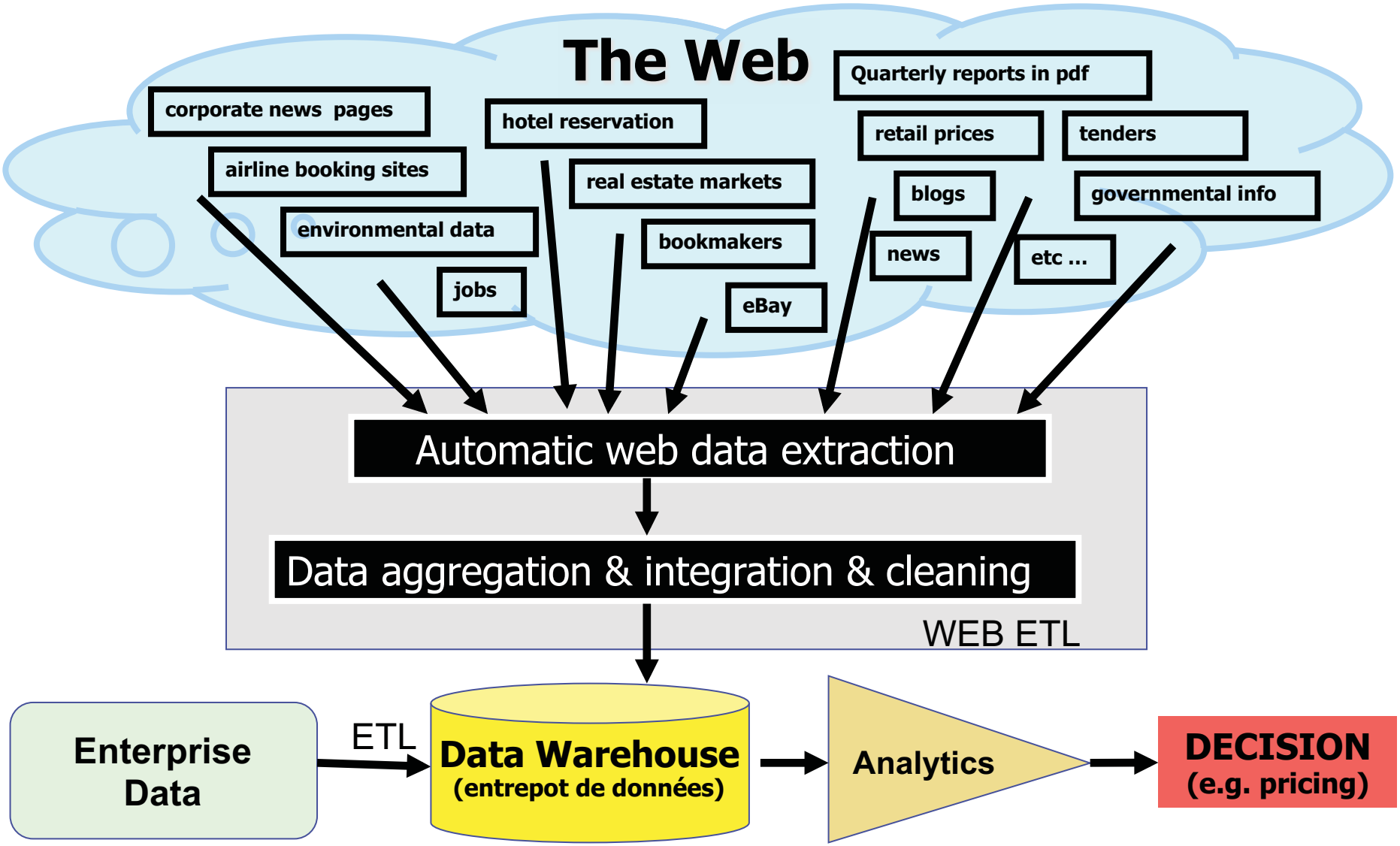
- **Road Construction Company:** Find new public tenders ("appels d'offre")

- **Hedge Fund** ("fonds de placement"): Obtain recent house price changes from real-estate agent's Web pages before the weekly index is published. Anticipating the Consumer price index (index des prix à la consommation).

- **Governmental/Policy Making ….**


The Web
corporate news pages
airline booking sites
environmental data
jobs
hotel reservation
real estate markets
bookmakers
eBay
Quarterly reports in pdf
retail prices
tenders
blogs
governmental info
news
etc ...


Enterprise Data
ETL
Data Warehouse
(entrepot de données)
Analytics
DECISION
(e.g. pricing)


The Web
corporate news pages
airline booking sites
environmental data
jobs
hotel reservation
real estate markets
bookmakers
eBay
Quarterly reports in pdf
retail prices
blogs
news
tenders
governmental info
etc ...
Automatic web data extraction
Data aggregation & integration & cleaning
WEB ETL
Enterprise Data
ETL
Data Warehouse
(entrepot de données)
Analytics
DECISION
(e.g. pricing)

Marketing & Business Intelligence








PIRELLI

Marketing Department

goulet d'étranglement

Oracle 9

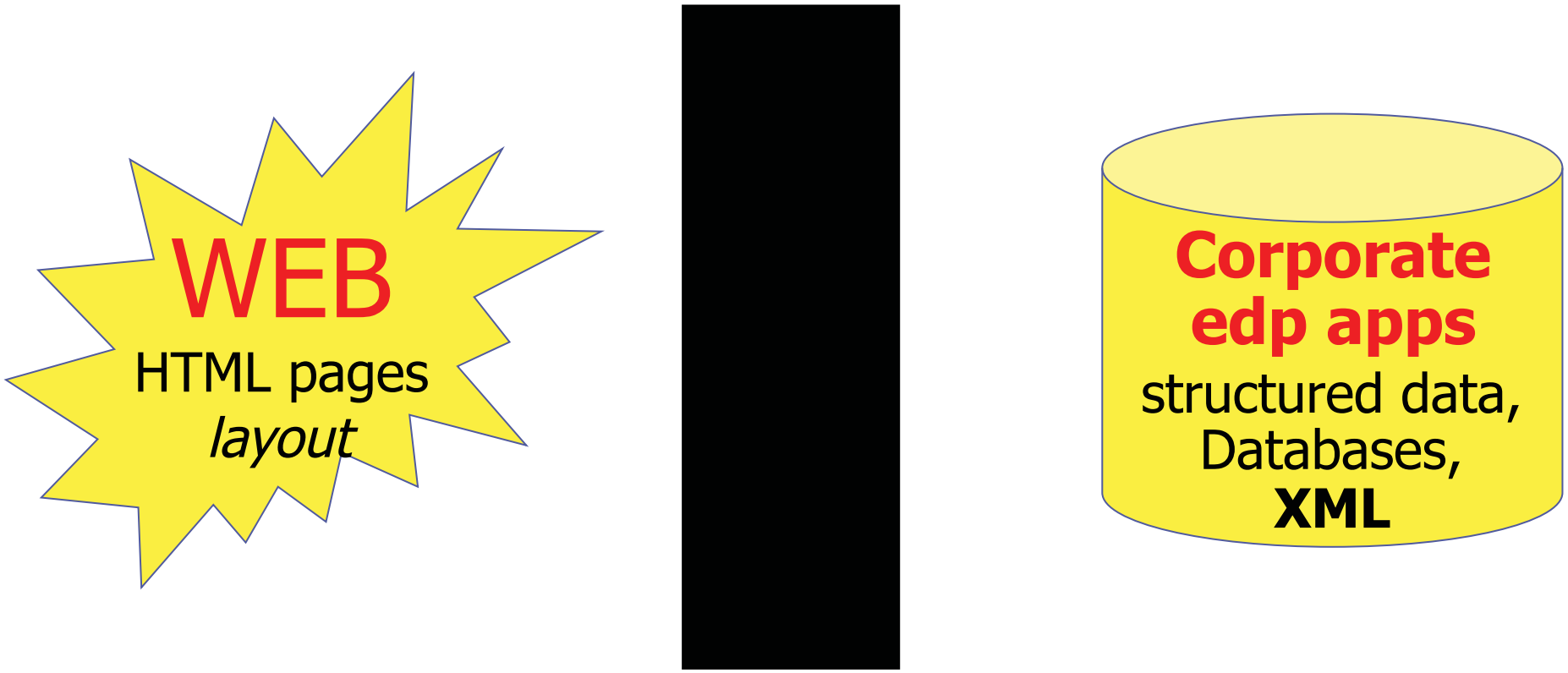
entrepot
de données

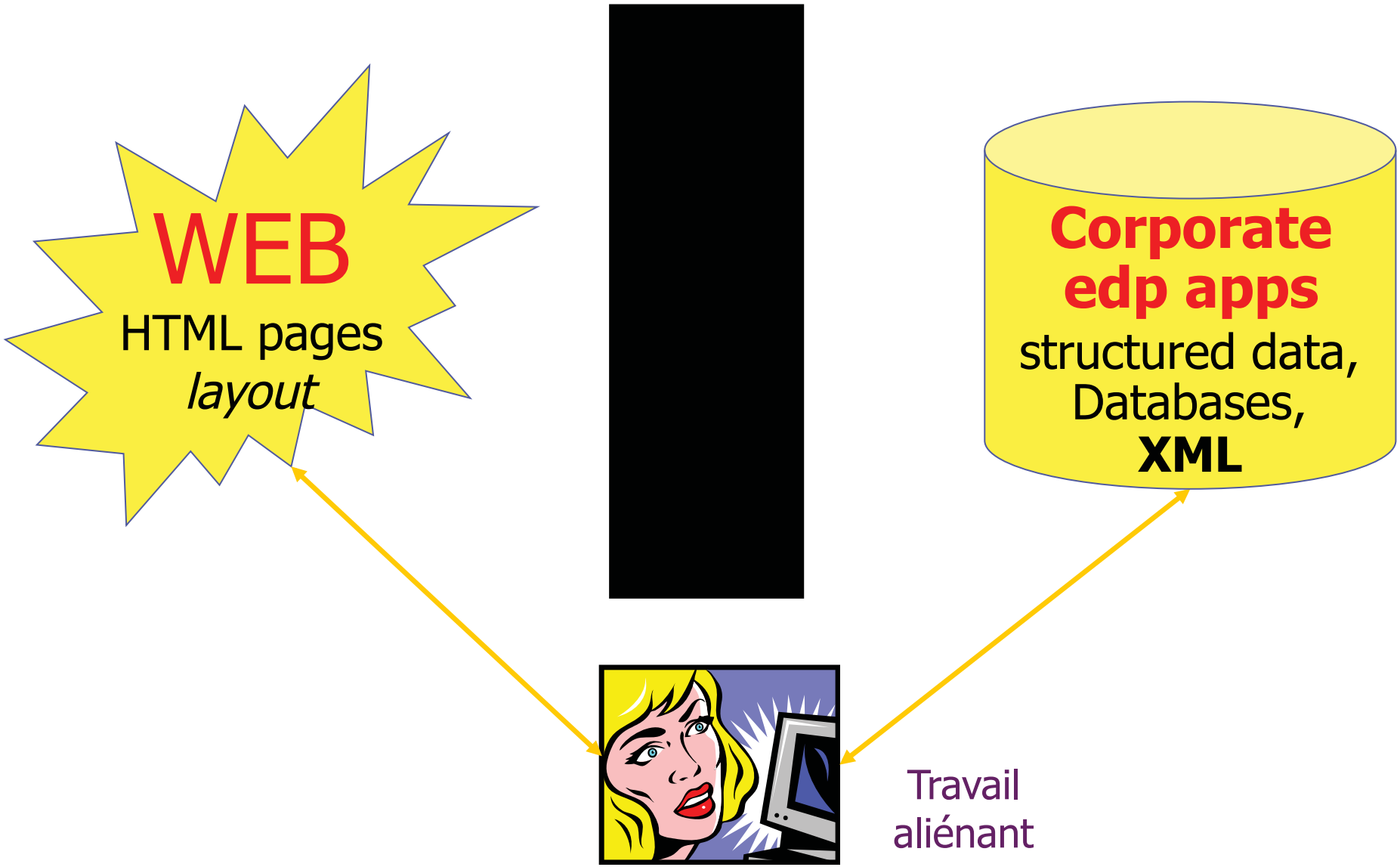
Business Objects report

BI Tool

The Wall

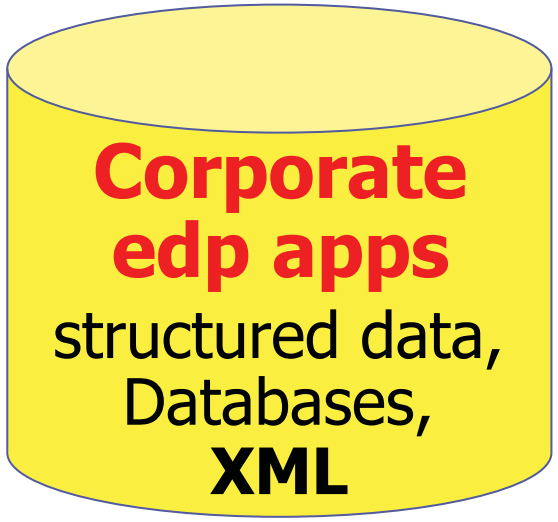
Problem: Make web contents accessible to electronic data processing




WEB
HTML pages
layout
Corporate
edp apps
structured data,
Databases,
XML
Travail
aliénant

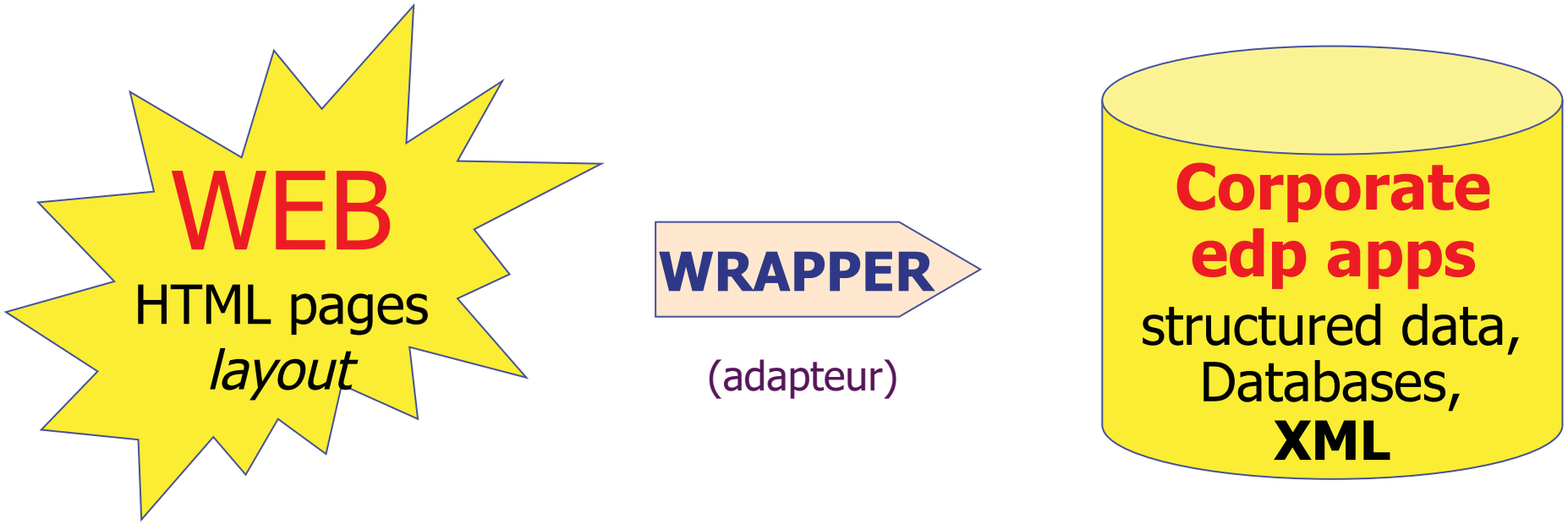
Web wrapping

Goal: Make web contents accessible to electronic data processing

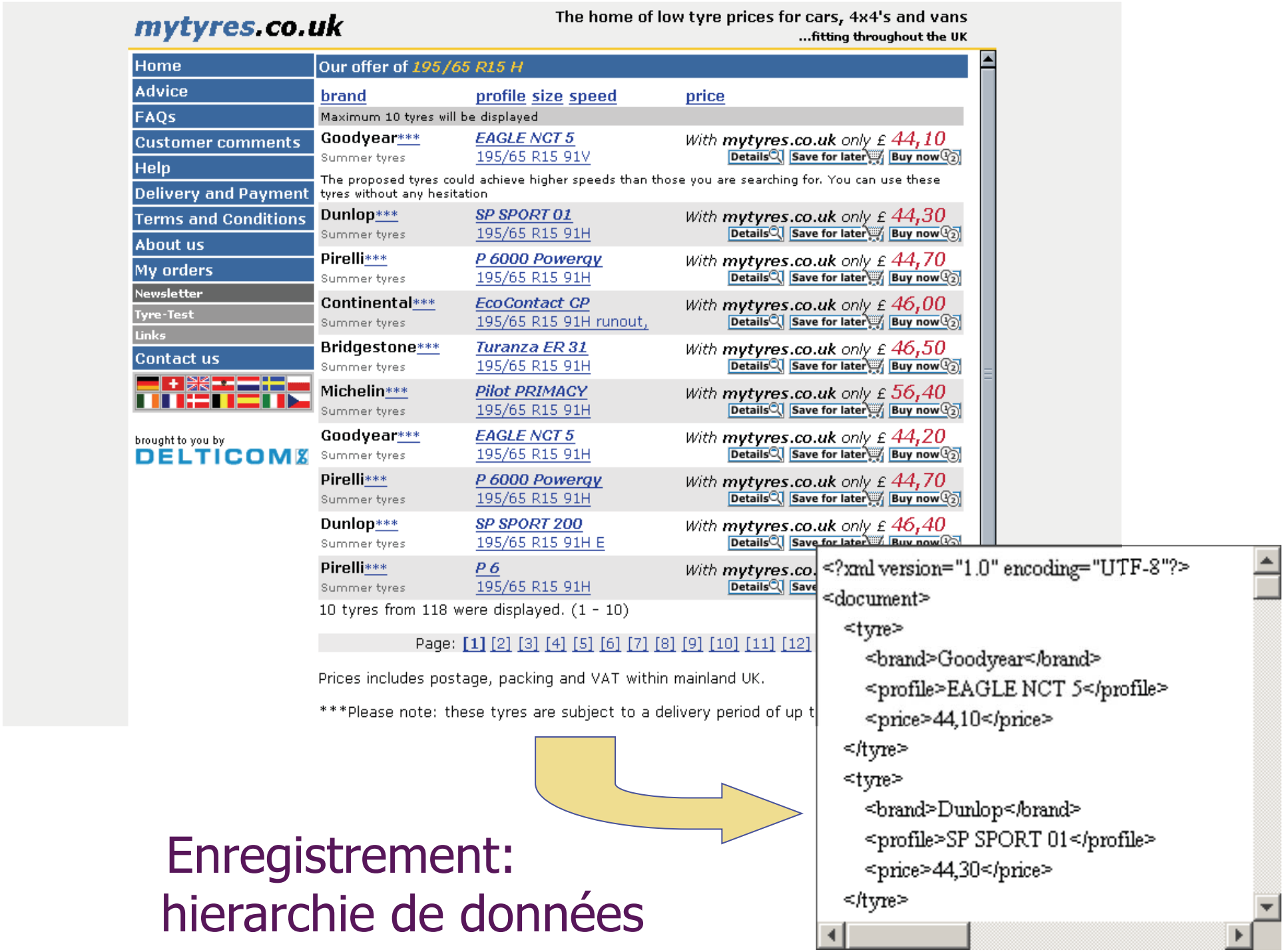


Web wrapping

Goal: Make web contents accessible to electronic data processing

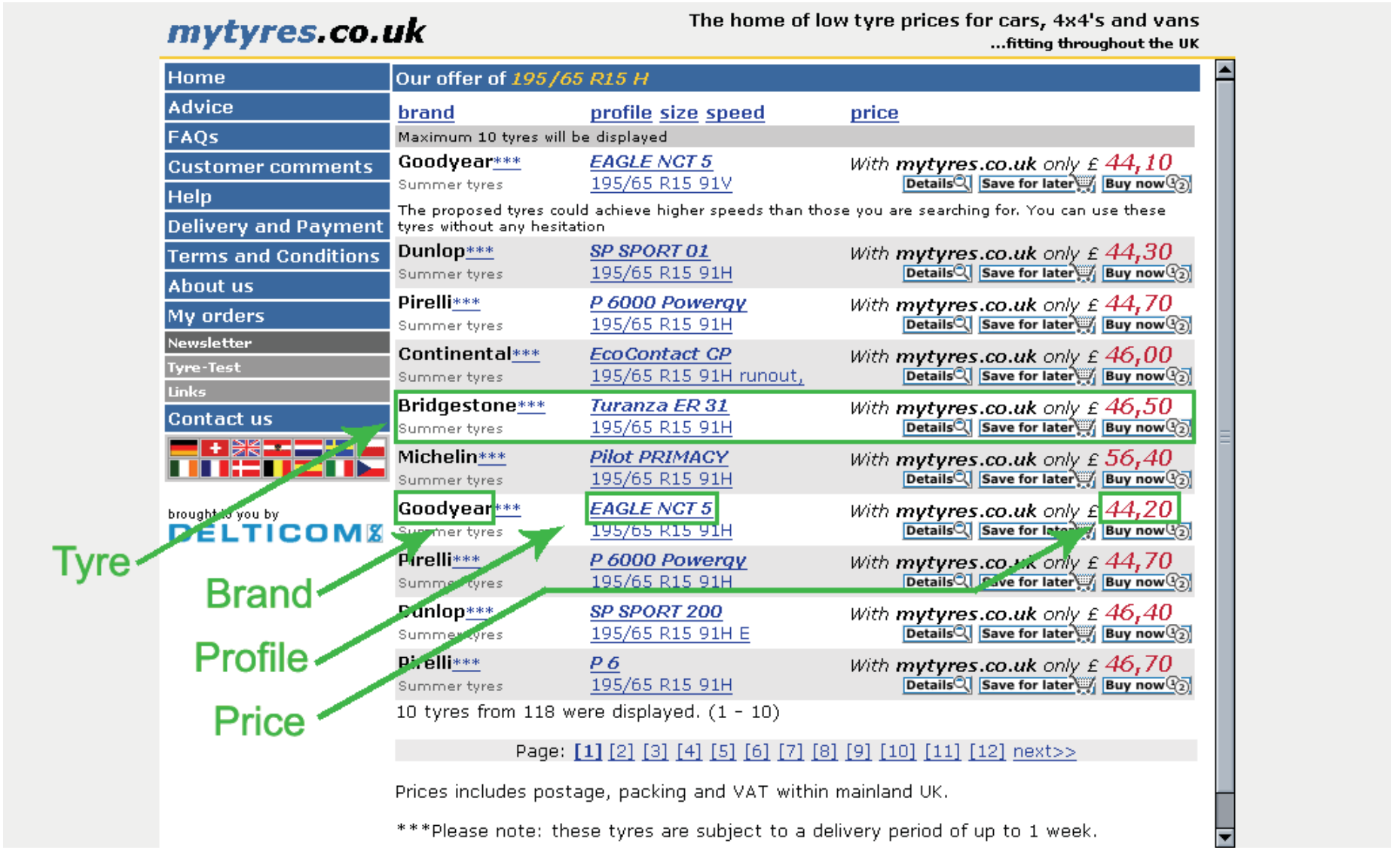


Wrappers: **HTML**→select → extract → annotate →**XML**



Enregistrement: hierarchie de données

Patterns:


mytyres.co.uk
The home of low tyre prices for cars, 4x4's and vans
...fitting throughout the UK
Home
Advice
FAQs
Customer comments
Help
Delivery and Payment
Terms and Conditions
About us
My orders
Newsletter
Tyre-Test
Links
Contact us
brought to you by
DELTICOM
Our offer of 195/65 R15 H
brand
profile
size
speed
price
Maximum 10 tyres will be displayed
Goodyear*** EAGLE NCT 5 Summer tyres 195/65 R15 91V With mytyres.co.uk only £ 44,10 Details Save for later Buy now
The proposed tyres could achieve higher speeds than those you are searching for. You can use these tyres without any hesitation
Dunlop*** SP SPORT 01 Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 44,30 Details Save for later Buy now
Pirelli*** P 6000 Powergy Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 44,70 Details Save for later Buy now
Continental*** EcoContact CP Summer tyres 195/65 R15 91H runout, With mytyres.co.uk only £ 46,00 Details Save for later Buy now
Bridgestone*** Turanza ER 31 Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 46,50 Details Save for later Buy now
Michelin*** Pilot PRIMACY Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 56,40 Details Save for later Buy now
Goodyear*** EAGLE NCT 5 Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 44,20 Details Save for later Buy now
Pirelli*** P 6000 Powergy Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 44,70 Details Save for later Buy now
Dunlop*** SP SPORT 200 Summer tyres 195/65 R15 91H E With mytyres.co.uk only £ 46,40 Details Save for later Buy now
Pirelli*** P 6 Summer tyres 195/65 R15 91H With mytyres.co.uk only £ 46,70 Details Save for later Buy now
10 tyres from 118 were displayed. (1 - 10)
Page: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] next>>
Prices includes postage, packing and VAT within mainland UK.
***Please note: these tyres are subject to a delivery period of up to 1 week.
Tyre
Brand
Profile
Price

Different approaches in the past

- **Programming** (Java, Perl, WebL, SQL+...)
  - very complicated & boring & expensive
  - testing very difficult
- **Simple Screen scrapers** ("gratte-écran")
  - no complex data structures extracted

- **Wrapper induction** (apprentissage d'adapteurs)
  - requires larger amounts of sample data
  - precision often not satisfactory
  - current systems text-based (not tree-based)

Different approaches in the past

- **Programming** (Java, Perl, WebL, SQL+...)
  - very complicated & boring & expensive
  - testing very difficult
- **Simple Screen scrapers**
  - no complex data structures extracted
- **Wrapper induction**
  - requires larger amounts of sample data
  - accuracy not satisfactory in all situations
  - current systems text-based (not tree-based)

Modern Solutions

- **Semi-automatic tool (outils)** 

  - based on solid theory
  - modular knowledge representation
  - easy to use
  - commercial product since 2002

- **Fully automated extraction** 

  - for specific application domains
  - extracts from 1000s of websites
  - current research

Talk Outline

- Motivation: need of information extraction
- Logical foundations of information extraction
- The Lixto Visual Wrapper
- The Diadem Project

Web documents are trees !

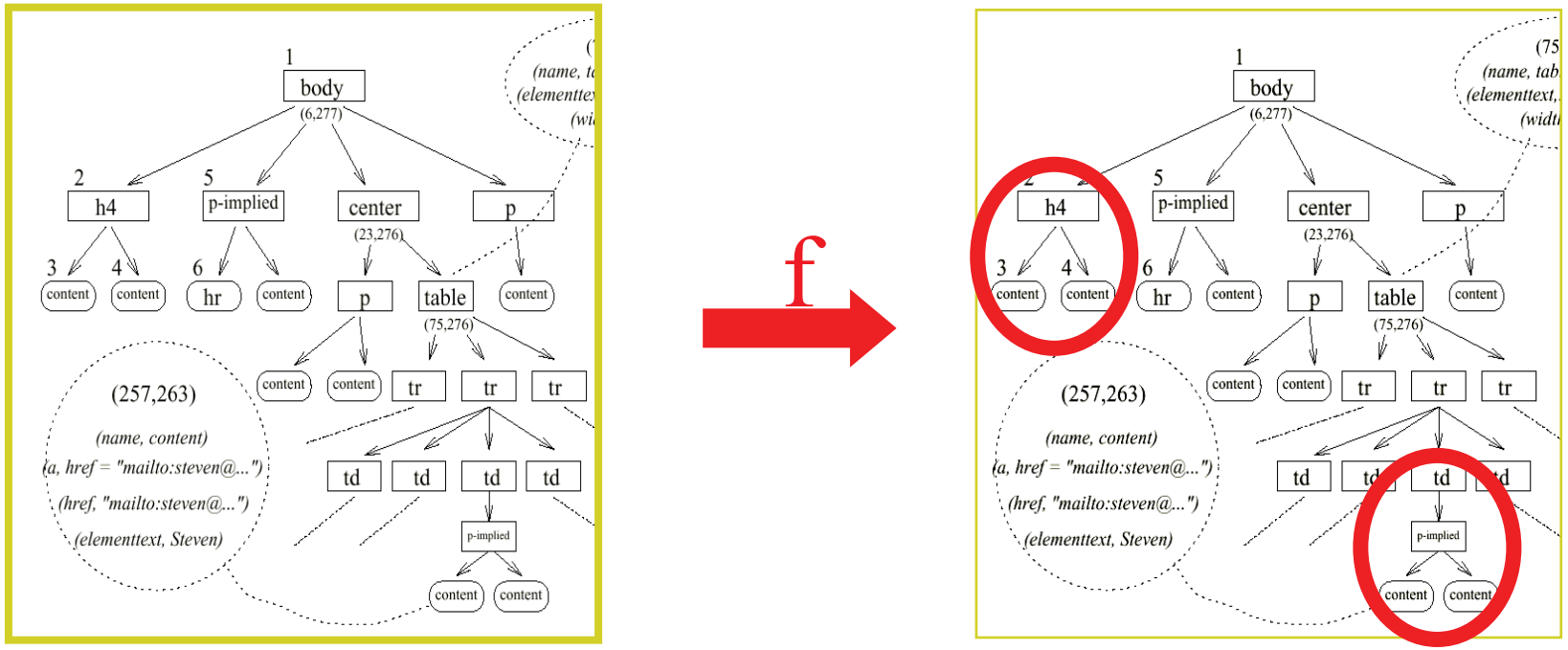
HTML: Hypertext Markup Language

XML: Extensible Markup Language

HTML, XML: Context free* languages. Represent a document by its parse tree (arbre syntaxique).

HTML Content Extractor

Function f: HTML Parse tree → Subtrees



Leaves of subtrees are among leaves of orig. tree

The Essence of Web Wrapping ?

Functional view: Wrapper defines functions f

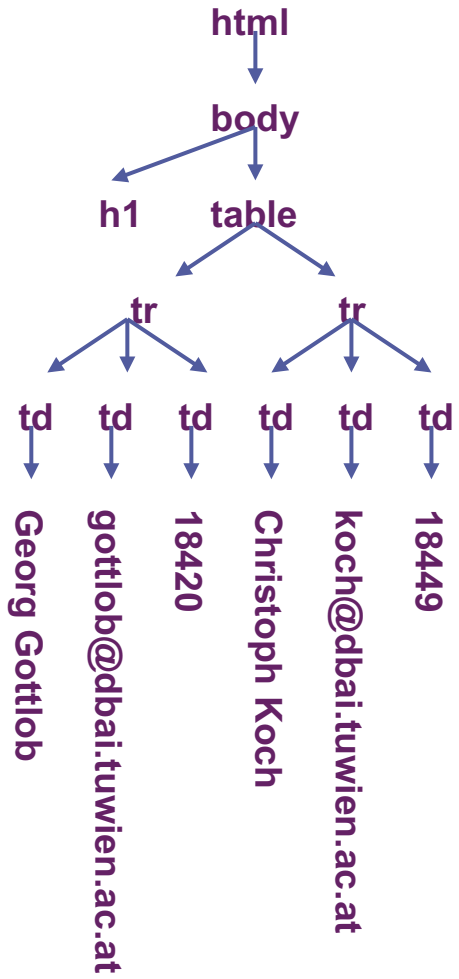
$$f: \text{Tree} \rightarrow \mathcal{P}(\text{Tree})$$
$$t \rightarrow T \subseteq \text{subtrees}(t)$$

Equivalent logical view:

Wrapper defines monadic predicates P over the nodes (arbre dom) **of each input document**

A HTML page

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html> <body>
<h1>People @ DBAI</h1>
<table border="1" cellpadding="3" cellspacing="1">
   <tr> <td>Georg Gottlob</td>
        <td>gottlob@dbai.tuwien.ac.at</td>
        <td>18420</td>
   </tr>
   <tr> <td>Christoph Koch</td>
        <td>koch@dbai.tuwien.ac.at</td>
        <td>18449</td>
   </tr>
</table>
</body> </html>
```

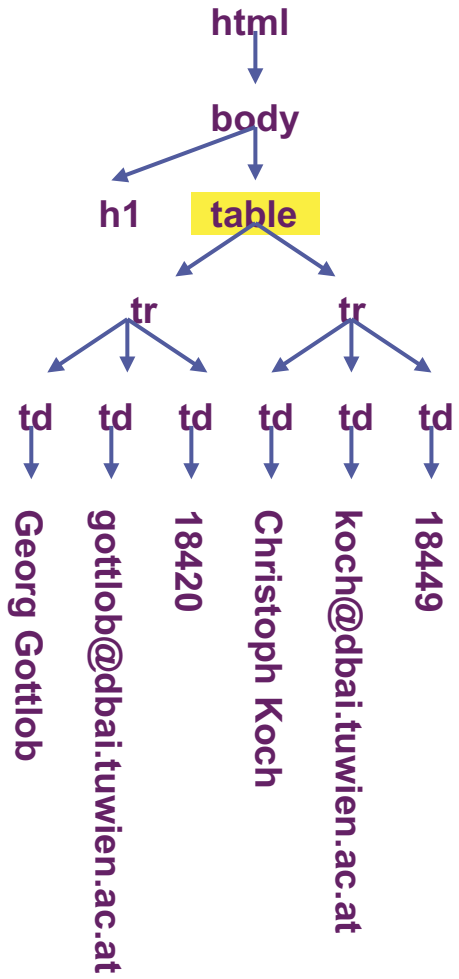


People @ DBAI

Georg Gottlob	gottlob@...	18420
Christoph Koch	koch@...	18449

Predicate *employeetable*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html> <body>
<h1>People @ DBAI</h1>
<table border="1" cellpadding="3" cellspacing="1">
   <tr> <td>Georg Gottlob</td>
        <td>gottlob@dbai.tuwien.ac.at</td>
        <td>18420</td>
   </tr>
   <tr> <td>Christoph Koch</td>
        <td>koch@dbai.tuwien.ac.at</td>
        <td>18449</td>
   </tr>
</table>
</body> </html>
```



People @ DBAI

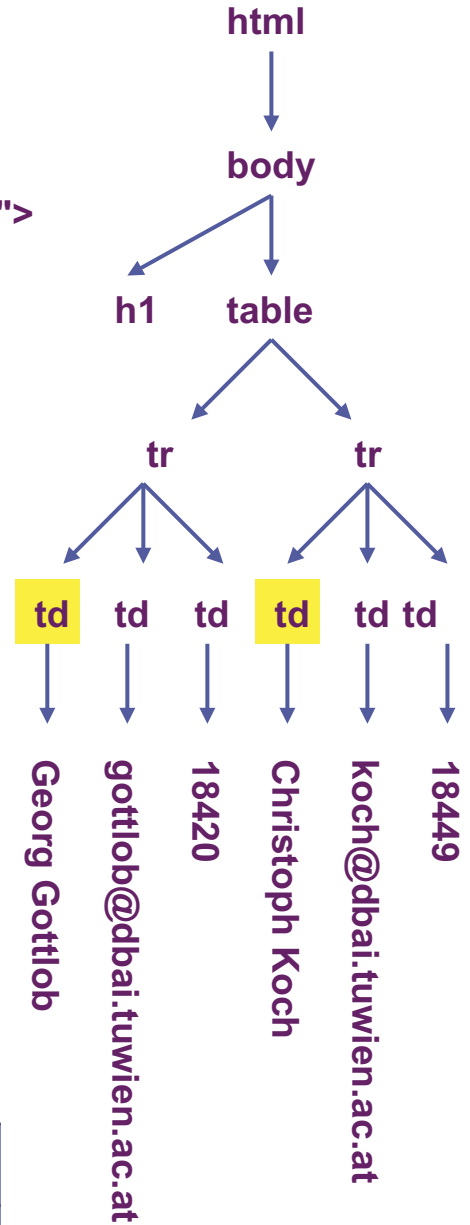
Georg Gottlob	gottlob@...	18420
Christoph Koch	koch@...	18449

Predicate *employee*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html> <body>
<h1>People @ DBAI</h1>
<table border="1" cellpadding="3" cellspacing="1">
   <tr> <td>Georg Gottlob</td>
        <td>gottlob@dbai.tuwien.ac.at</td>
        <td>18420</td>
   </tr>
   <tr> <td>Christoph Koch</td>
        <td>koch@dbai.tuwien.ac.at</td>
        <td>18449</td>
   </tr>
</table>
</body> </html>
```

People @ DBAI

Georg Gottlob	gottlob@...	18420
Christoph Koch	koch@...	18449

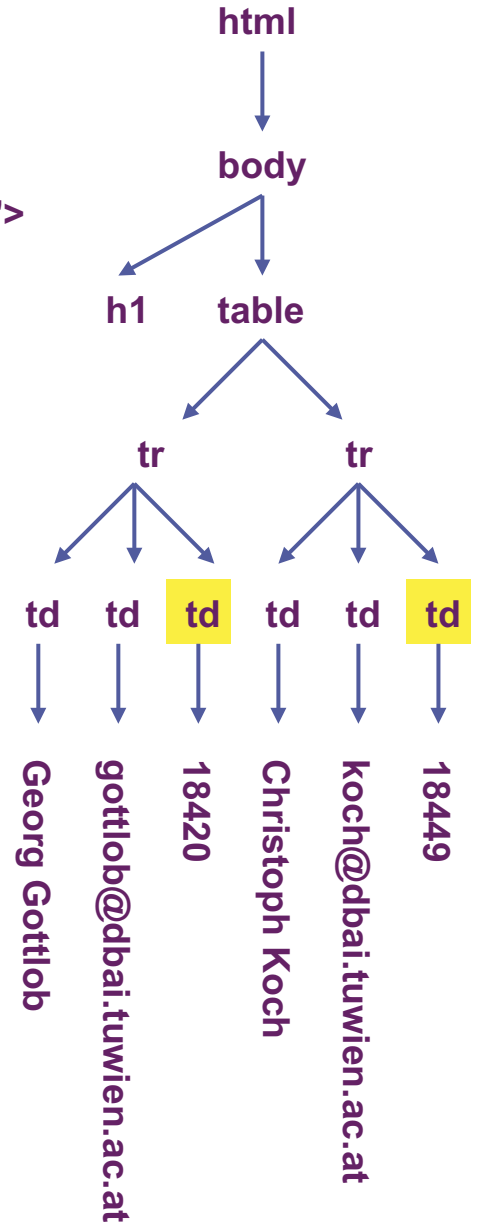


Predicate *phone*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html> <body>
<h1>People @ DBAI</h1>
<table border="1" cellpadding="3" cellspacing="1">
   <tr> <td>Georg Gottlob</td>
        <td>gottlob@dbai.tuwien.ac.at</td>
        <td>18420</td>
   </tr>
   <tr> <td>Christoph Koch</td>
        <td>koch@dbai.tuwien.ac.at</td>
        <td>18449</td>
   </tr>
</table>
</body> </html>
```

People @ DBAI

Georg Gottlob	gottlob@…	18420
Christoph Koch	koch@…	18449



Expressiveness Yardstick: MSO

- MSO captures exactly the essence of data extraction:
  - Define sets of nodes of a document

- Expressiveness, complexity, semantics well understood:
  - ✓ MSO over trees: perfect logical semantics
  - ✓ MSO over trees: high expressive power (tree automata)
  - ✓ MSO over trees: low data complexity

- Drawbacks: - hard to use, no visual specification,
  - high query complexity (cpl. de requetes)

  (➔ bad scalability, mauvais passage à l'échelle).

MSO on strings and trees

Rich theory:

- Büchi: MSO = REG over strings (chaînes de caractères)
- Thatcher and Wright, Rabin:

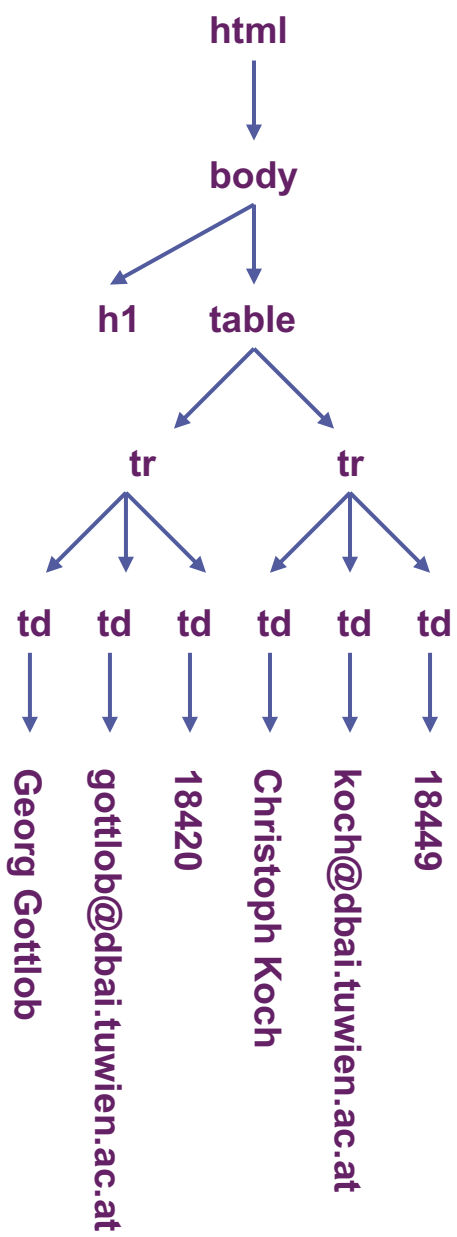
  MSO = REG over ranked trees (arbres bornés)

  = tree automata
- Brüggemann-Klein/Wood/Murata:

  MSO = REG over unranked trees
- Neven & Schwentick: Unranked Query Automata
- Courcelle: MSO in LinTime on tree-like structures

  (treewidth <= k, data complexity)

Ordered Trees as finite structures

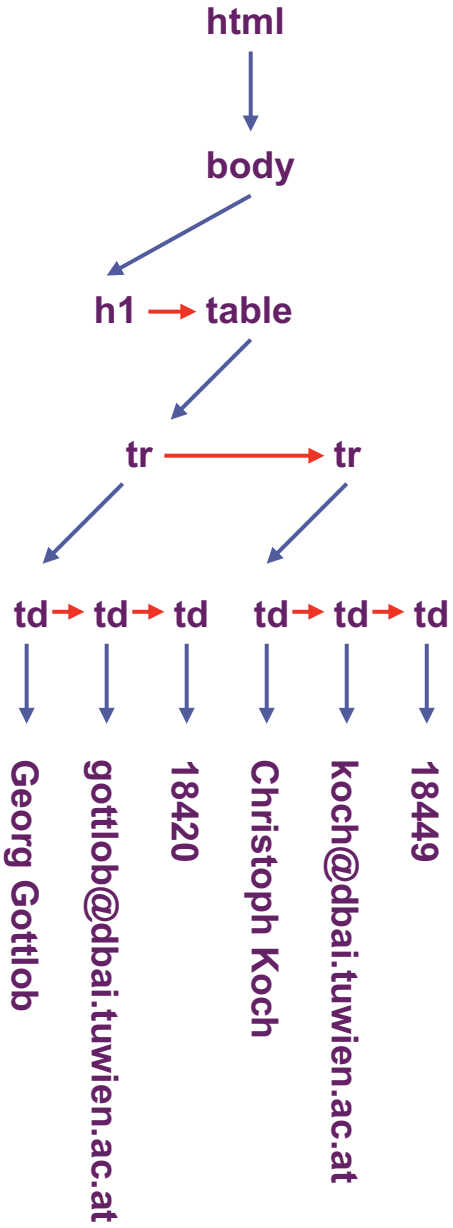


firstchild

nextsibling

$label_{h1}()$
$label_{td}()$
...

root()
leaf()

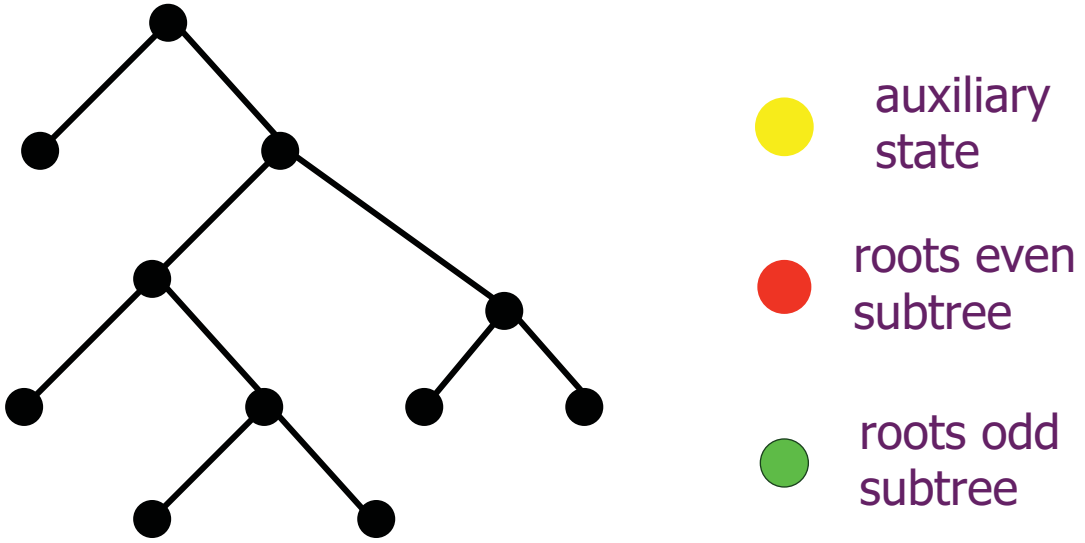


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

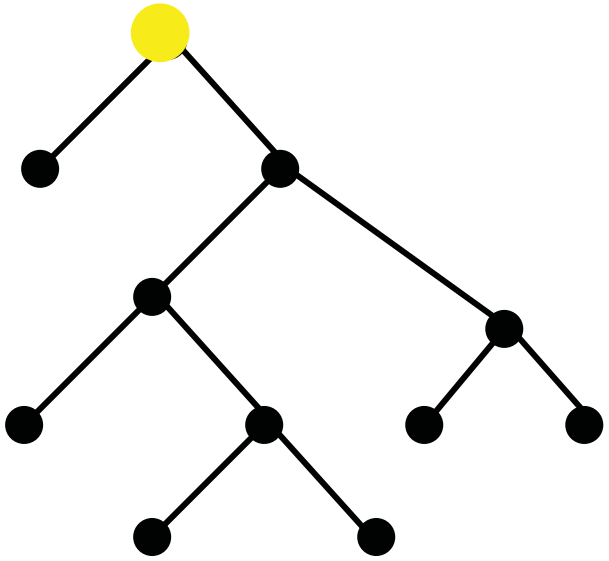


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

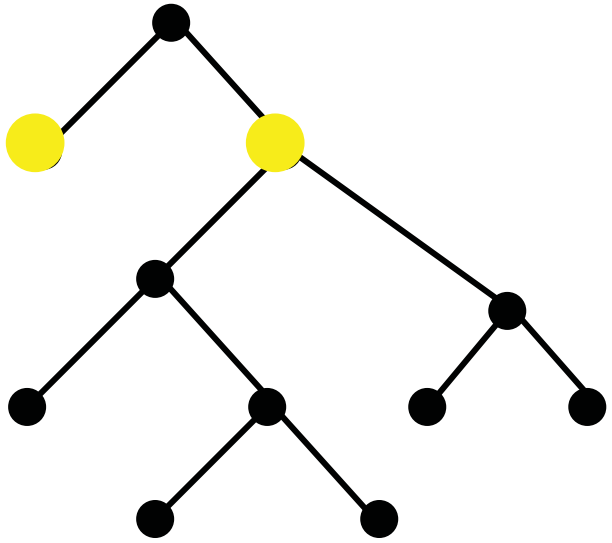


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

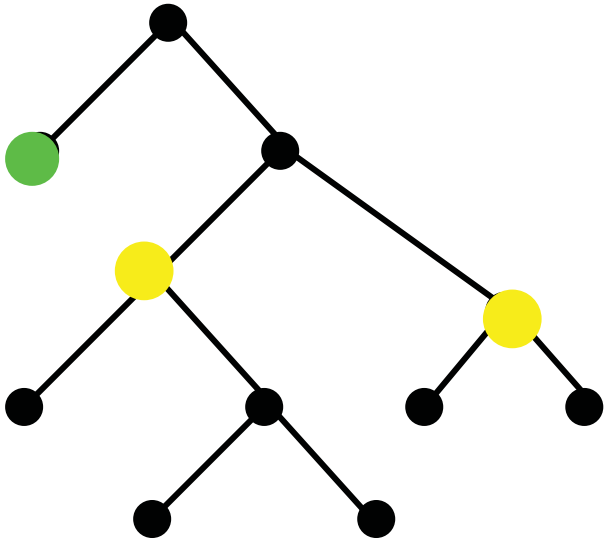


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

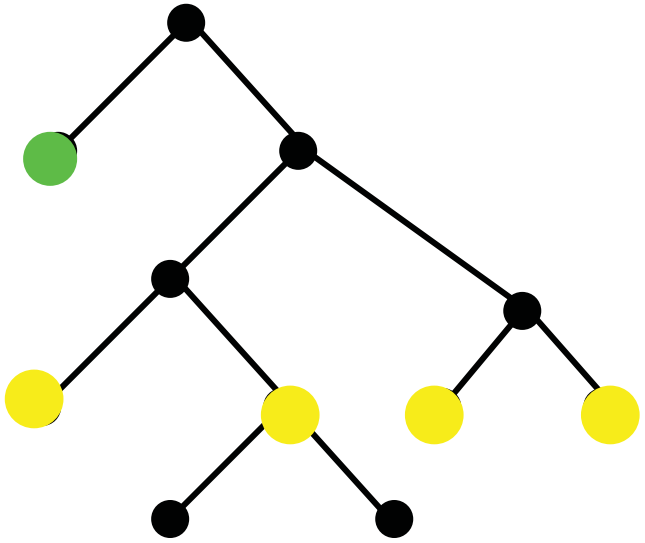


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

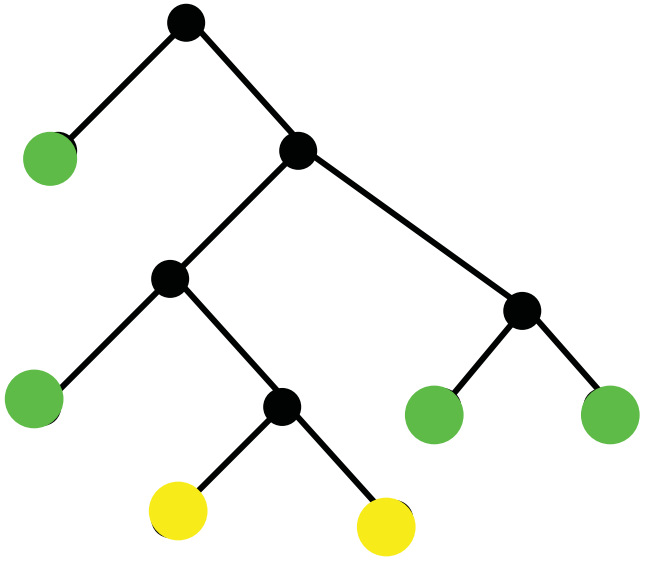


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

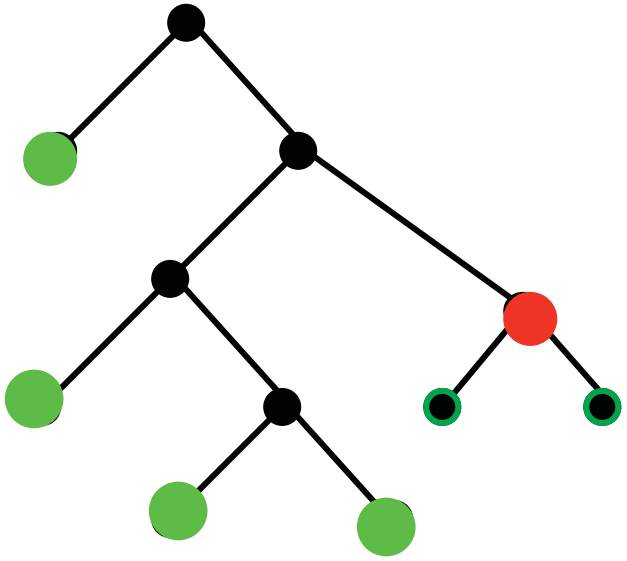


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

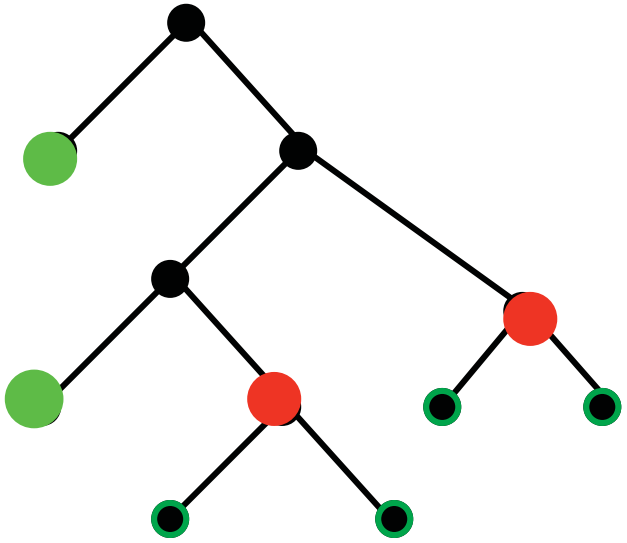


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

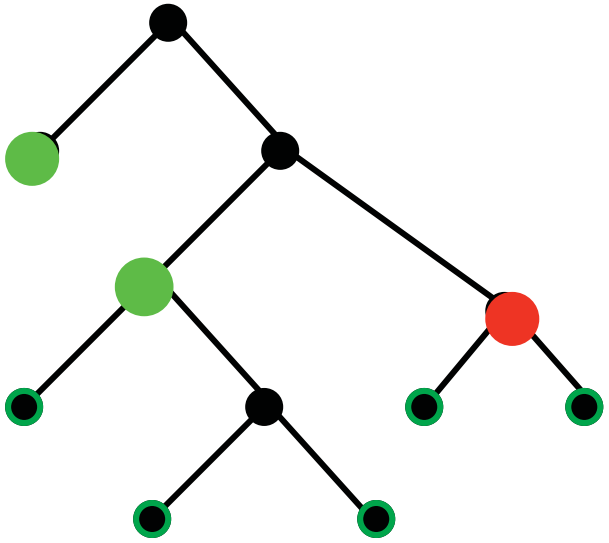


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

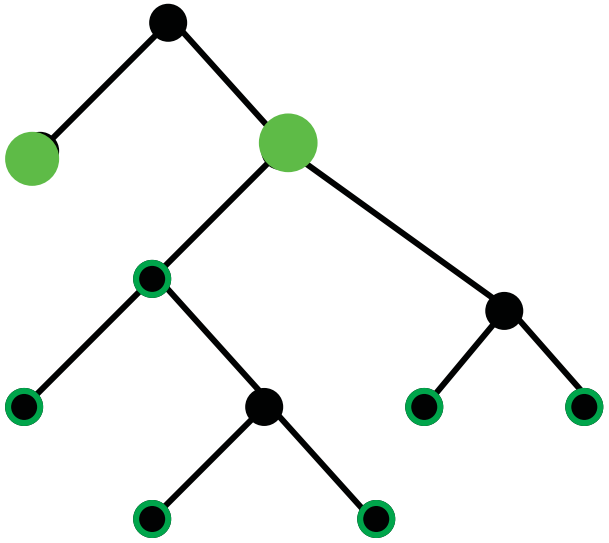


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

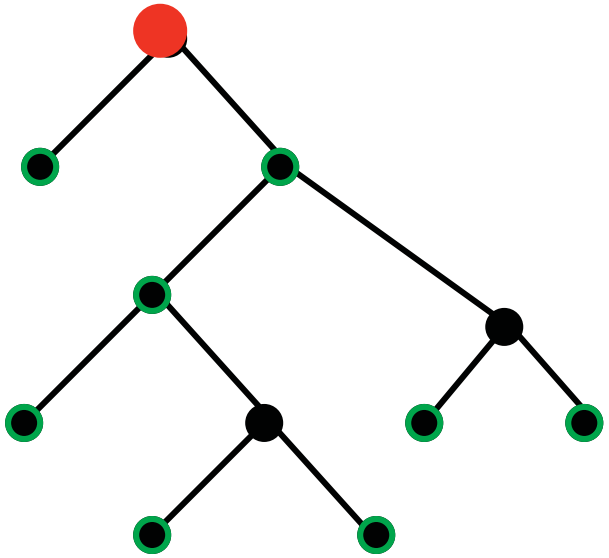


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:

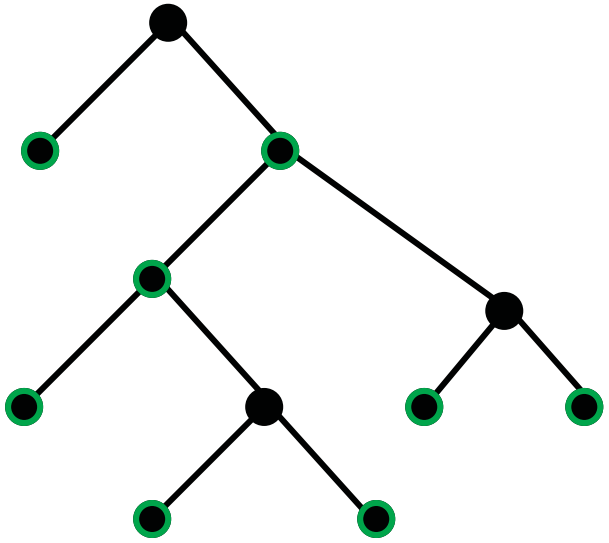


MSO over Trees

Extract from a binary tree all roots of sub-trees with an odd number of leaves:

$$\exists S\ \forall x\ [\ S(u)\ \&\ (\ leaf(x) \rightarrow S(x))\ \&$$
$$\forall\ x,y,z\ (((firstchild(x,y)\ \&\ nextsibling(y,z)) \rightarrow$$
$$(S(x) \leftrightarrow \neg(S(y) \leftrightarrow S(z))))]$$

Tree automaton:



Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

||

Monadic Datalog

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

$=$

Monadic Datalog

$\cap$

Elog

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

=

Monadic Datalog

∩

Elog

∩

Lixto Visual Wrapper

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

=

Monadic Datalog

∩

Elog

∩

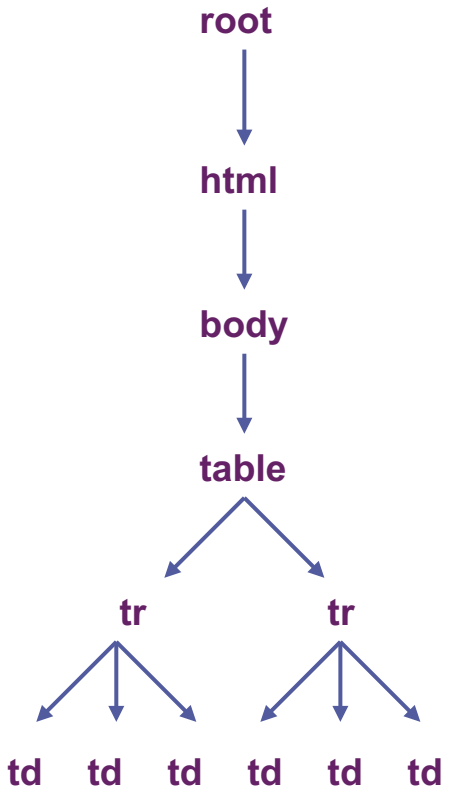
Lixto Visual Wrapper

∩

Lixto Suite

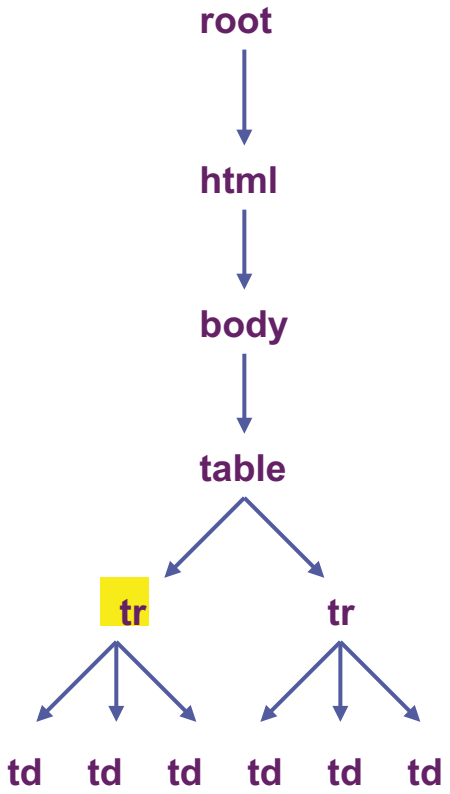
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```



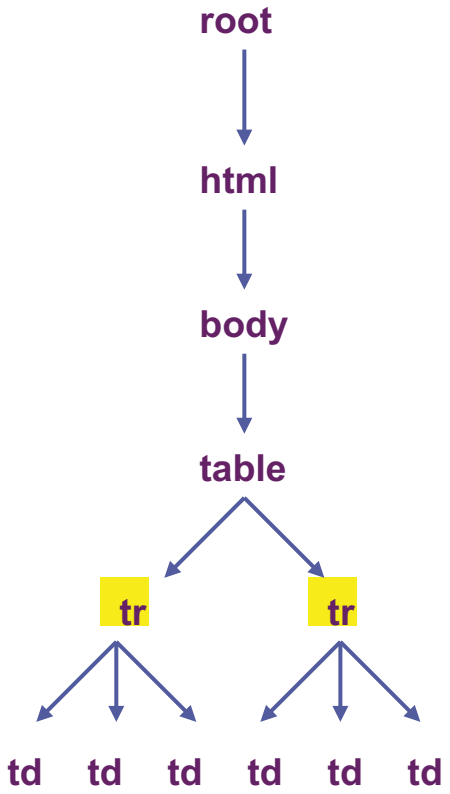
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
            firstchild(U,V), label[body](V),
            firstchild(V,W),label[table](W),
            firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```



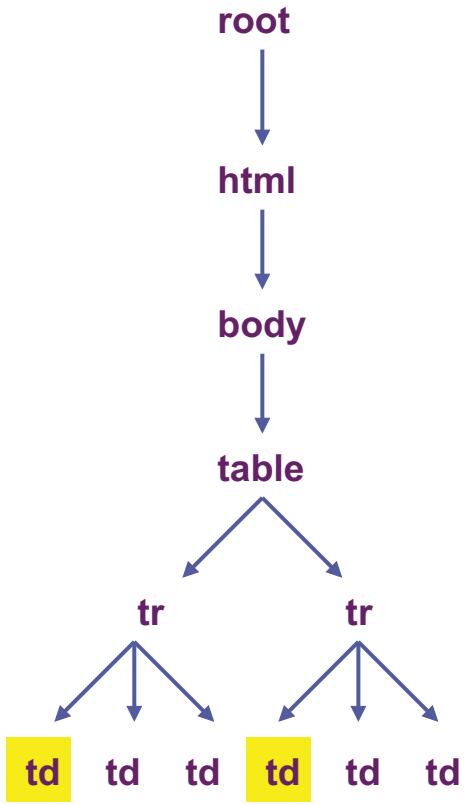
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).

name(X) :- entry(E), firstchild(E, X), label[td](X).

email(X) :- name(N), nextsibling(N, X), label[td](X).

phone(X) :- email(M), nextsibling(M, X), label[td](X).
```



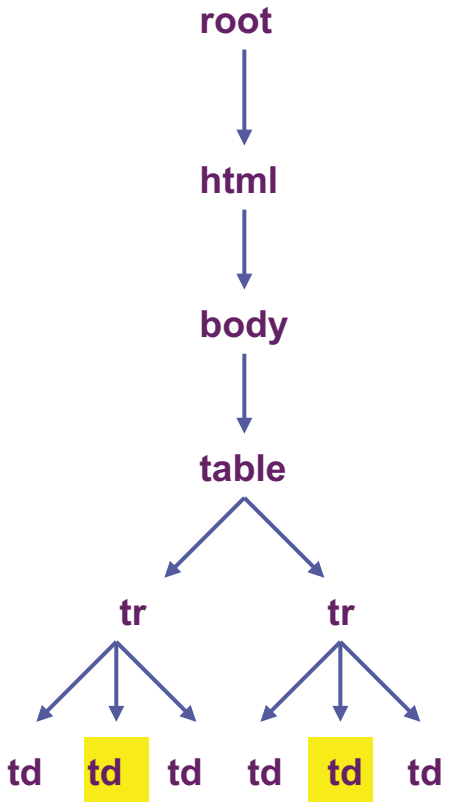
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```



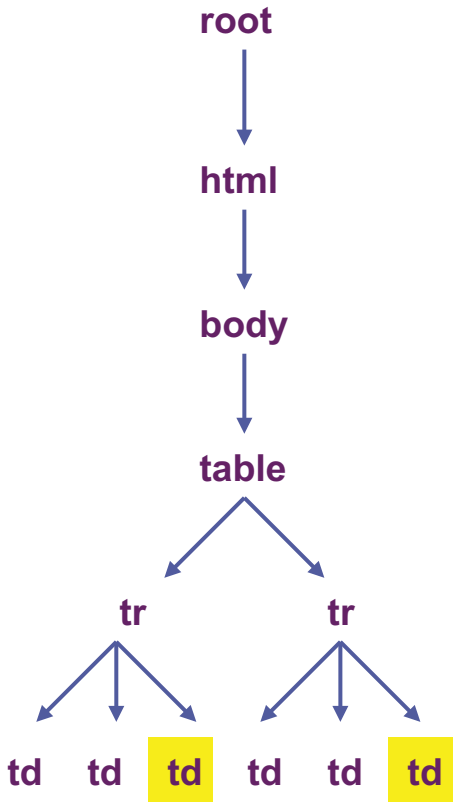
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```



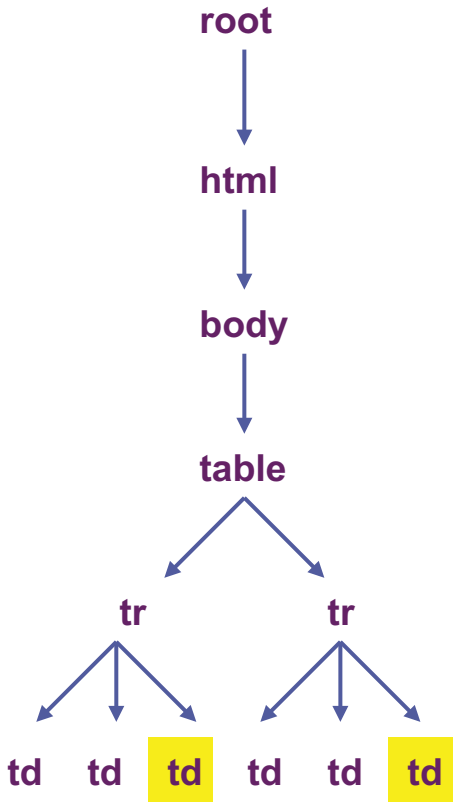
Monadic Datalog as a Wrapping Language

```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```

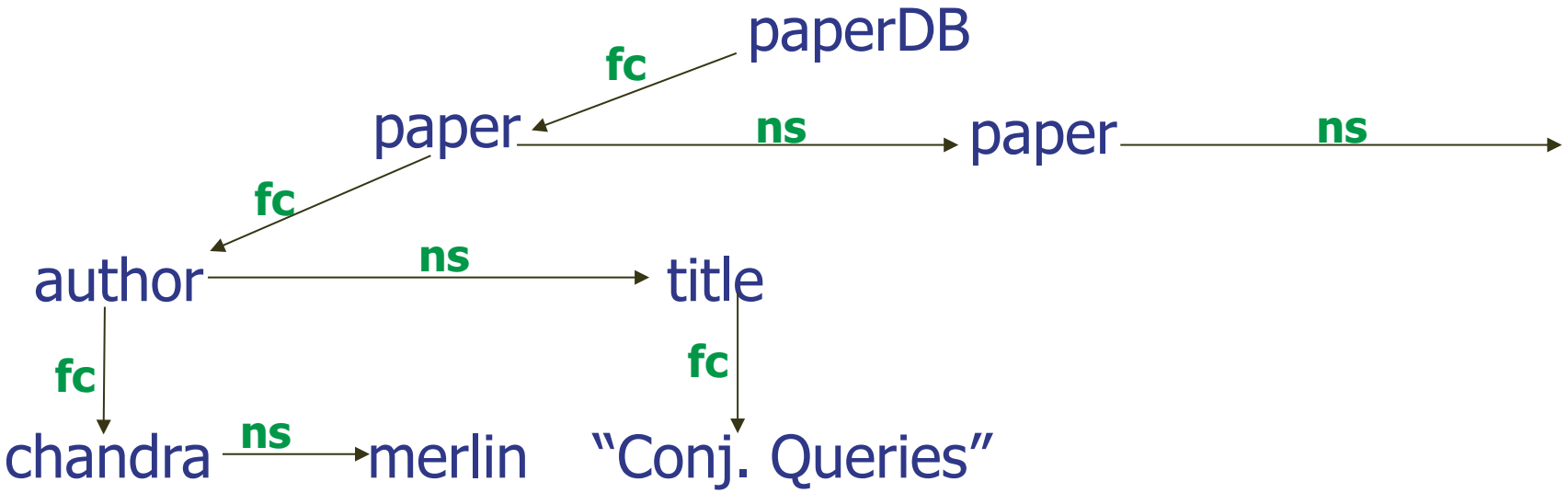


```
entry(X) :- root(R), firstchild(R,U), label[html](U),
                     firstchild(U,V), label[body](V),
                     firstchild(V,W),label[table](W),
                     firstchild(W,X), label[tr](X).
entry(X):- entry(Y), nextsibling(Y,X).
name(X) :- entry(E), firstchild(E, X), label[td](X).
email(X) :- name(N), nextsibling(N, X), label[td](X).
phone(X) :- email(M), nextsibling(M, X), label[td](X).
```

```
<?xml version="1.0"?>
<peopledb>
<entry> <name>Georg Gottlob</name>
        <email>gottlob@dbai.tuwien.ac.at</email>
        <phone>18420</phone>
</entry>
<entry> <name>Christoph Koch</name>
        <email>koch@dbai.tuwien.ac.at</email>
        <phone>18449</phone>
</entry>
</peopledb>
```



Monadic Datalog over XML



Select titles of articles authored by Chandra and Merlin

paper(X) ← root(R) & firstchild(R,X).
paper(X) ← paper(Y) & nextsibling(Y,X).

output(X)← paper(P) & firstchild(P,A) &
firstchild(A,Z) & label[Chandra](Z) &
nextsibling(Z,V) & label[Merlin](V) &
nextsibling(A,T) & firstchild(T,X).

How expressive is monadic Datalog?

It was known that over arbitrary structures:

- Monadic Datalog $\subseteq \Pi_1$-MSO
- Full Datalog = P (in presence of order)

Theorem [G. & Koch 2002]:

Over trees, monadic Datalog = MSO

A unary query is definable in MSO iff it is definable via a monadic datalog program.

How complex is Monadic Datalog?

Theorem [G. & Koch 2002]:

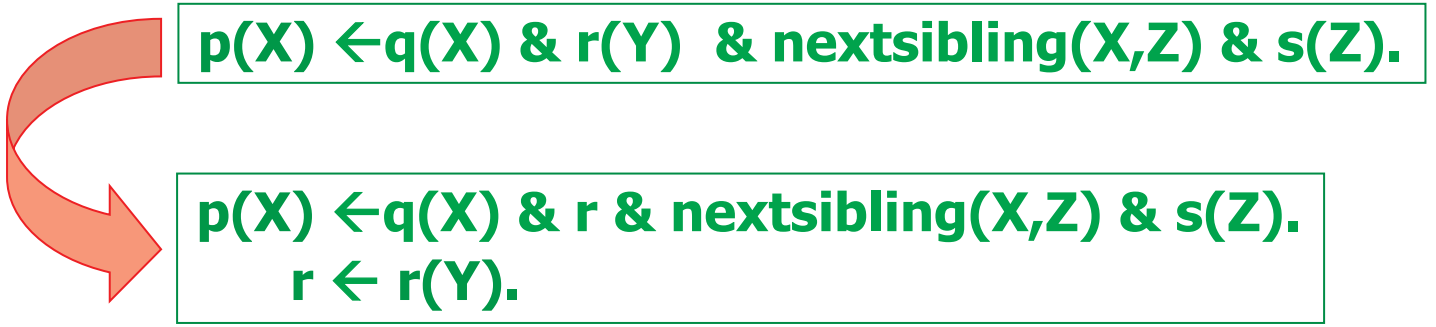
Monadic Datalog over trees has
combined complexity: $O(|data| * |query|)$

Query Complexity: P-complete and linear-time.

Proof idea:

1.) Transform datalog program + input tree in linear time into a "ground" propositional logic program (programme Datalog instancié)

- Exploit functional dependencies:
  nextsibling(X,Y) has only a linear number of ground instances: nextsibling(n_i,n_j), etc.
- Decouple independent atoms of rule bodies



2.) Execute ground program in linear time by using well-known algorithms: [Beeri&Bernstein][Dowling&Gallier] [Minoux]

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

$=$

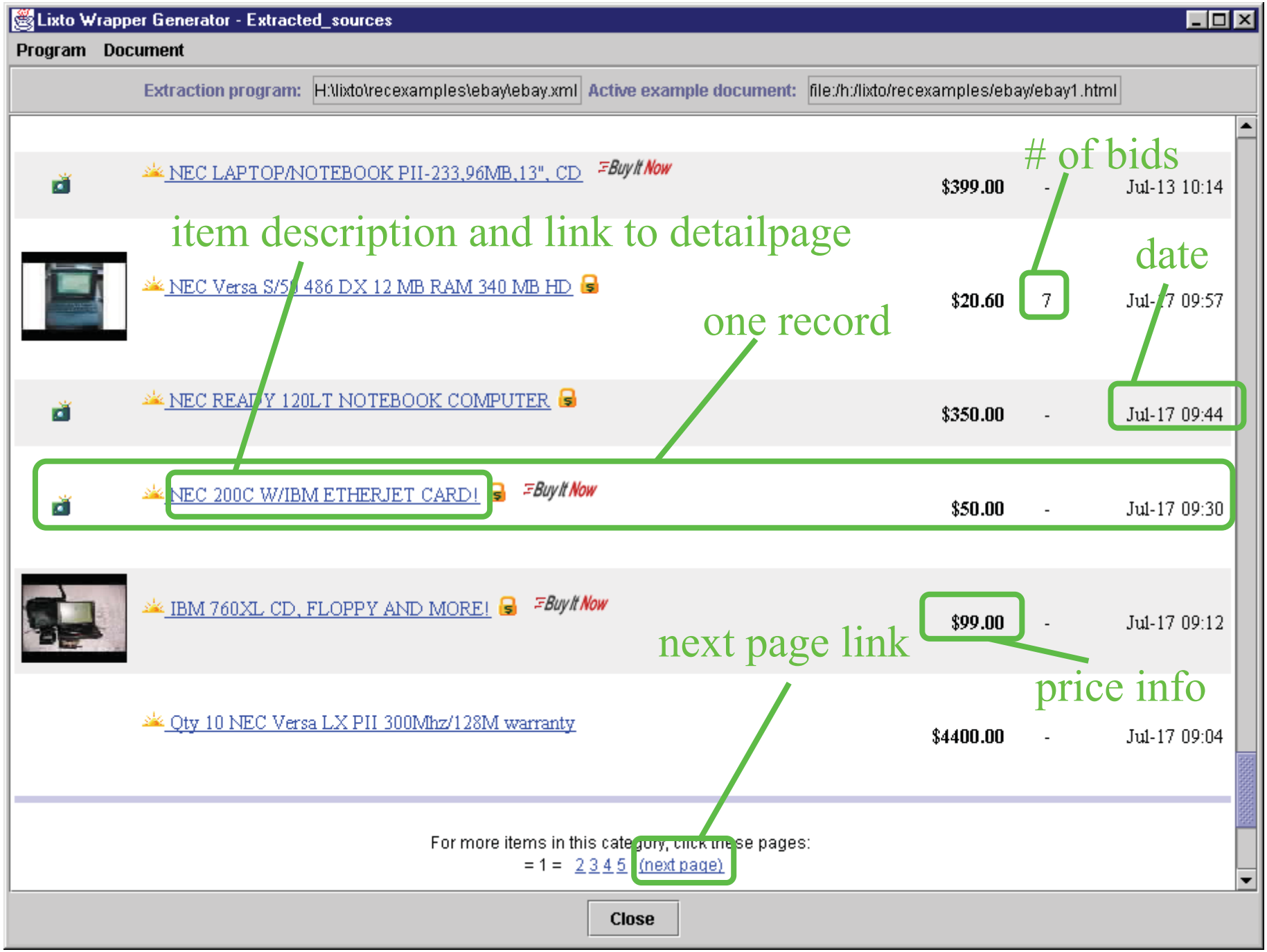
Monadic Datalog

$\in$

Elog

$\in$

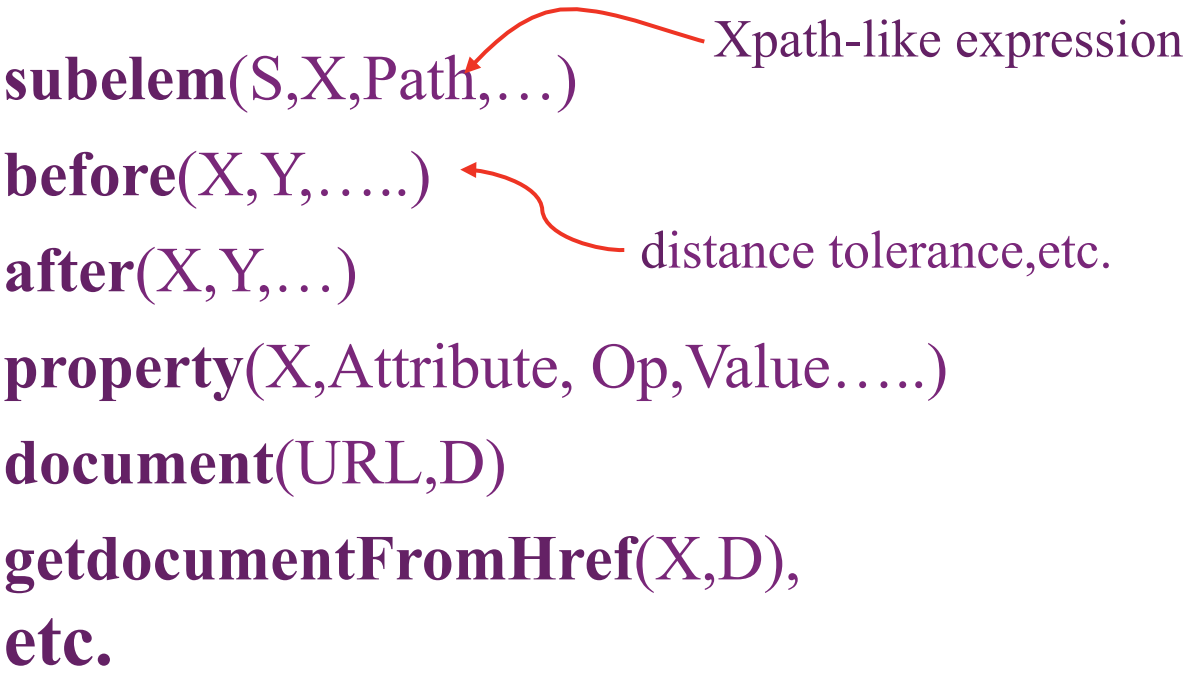
Lixto Visual Wrapper


Lixto Wrapper Generator - Extracted_sources
Program Document
Extraction program: H:\lixto\recexamples\ebay\ebay.xml
Active example document: file:/h:/lixto/recexamples/ebay/ebay1.html
NEC LAPTOP/NOTEBOOK PII-233,96MB,13", CD
Buy It Now
$399.00 - Jul-13 10:14
NEC Versa S/50 486 DX 12 MB RAM 340 MB HD
$20.60 7 Jul-17 09:57
NEC READY 120LT NOTEBOOK COMPUTER
$350.00 - Jul-17 09:44
NEC 200C W/IBM ETHERJET CARD!
Buy It Now
$50.00 - Jul-17 09:30
IBM 760XL CD, FLOPPY AND MORE!
Buy It Now
$99.00 - Jul-17 09:12
Qty 10 NEC Versa LX PII 300Mhz/128M warranty
$4400.00 - Jul-17 09:04
For more items in this category, click these pages:
= 1 = 2 3 4 5 (next page)
Close
item description and link to detailpage
one record
# of bids
date
next page link
price info

ELOG

[Baumgartner, Flesca, G. VLDB' 01]

Examples of Special predicates:



subelem(S,X,Path,…)

before(X,Y,…..)

after(X,Y,…)

property(X,Attribute, Op,Value…..)

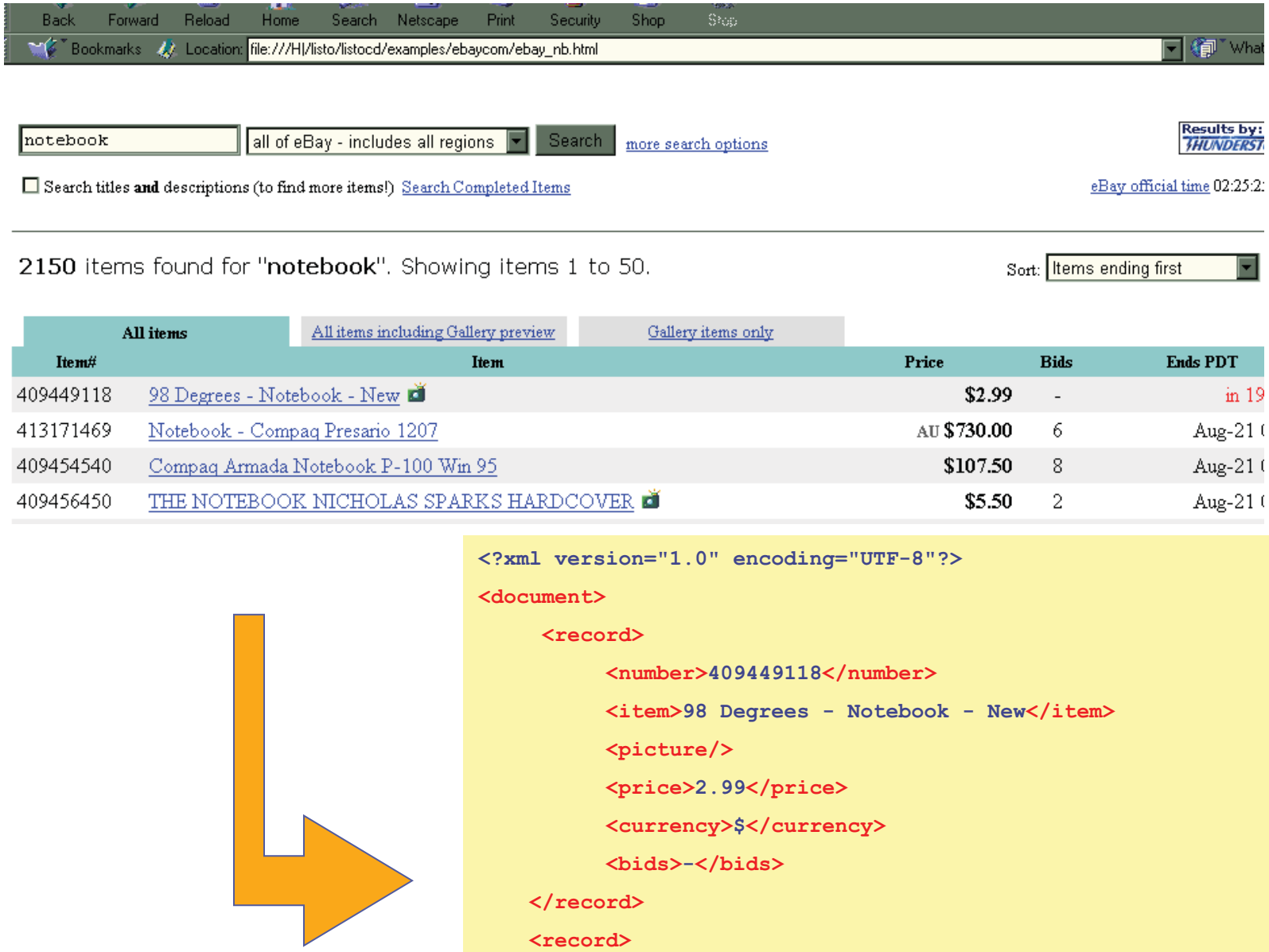
document(URL,D)

getdocumentFromHref(X,D),

etc.

Additional features:

Stratified negation,
string processing
ontological concepts "phonenumber(X)"
ranges: H(S,X) :- body(……..)[1,5]
object hierarchies




```
<?xml version="1.0" encoding="UTF-8"?>
<document>
     <record>
          <number>409449118</number>
          <item>98 Degrees - Notebook - New</item>
          <picture/>
          <price>2.99</price>
          <currency>$</currency>
          <bids>-</bids>
    </record>
    <record>
          <number>413171469</number>
          <item>Notebook - Compaq Presario 1207</item>
          <price>730.00</price>
          <currency>AU $</currency>
[...]
```

ELOG Program for eBay pages

```
tableseq(S, X) ← document("www.ebay.com/", S), subsq(S, (.body, []), (.table, []), (.table, []), X),
                 before(S, X, (.table, [(elementtext, item, [])], 1, 1, _, _), after(S, X, (.hr, []), 1, 1, _, _)

record(S, X) ← tableseq(_, S), subelem(S, .table, X)

itemnum(S, X) ← record(_, S), subelem(S, ⋆.td, X), notbefore(S, X, (.td, []), maxint)

itemdes(S, X) ← record(_, S), subelem(S, (⋆.td. ⋆ .content, [(a, , 0)], X)

price(S, X) ← record(_, S), subelem(S, (⋆.td, [(elementtext, Y, 1)]), X), valuta(Y)

bids(S, X) ← record(_, S), subelem(S, ⋆.td, X), before(S, X, (.td, []), 1, 30, Y, _), price(S, Y)

currency(S, X) ← price(_, S), subtext(S, Y, X), valuta(Y)

pricewc(S, X) ← price(_, S), subtext(S, [0 − 9]+, X)
```

Logic heaven

DB theory heaven

DB programming heaven

Application design heaven

MSO

$\equiv$

Monadic Datalog

$\cap$

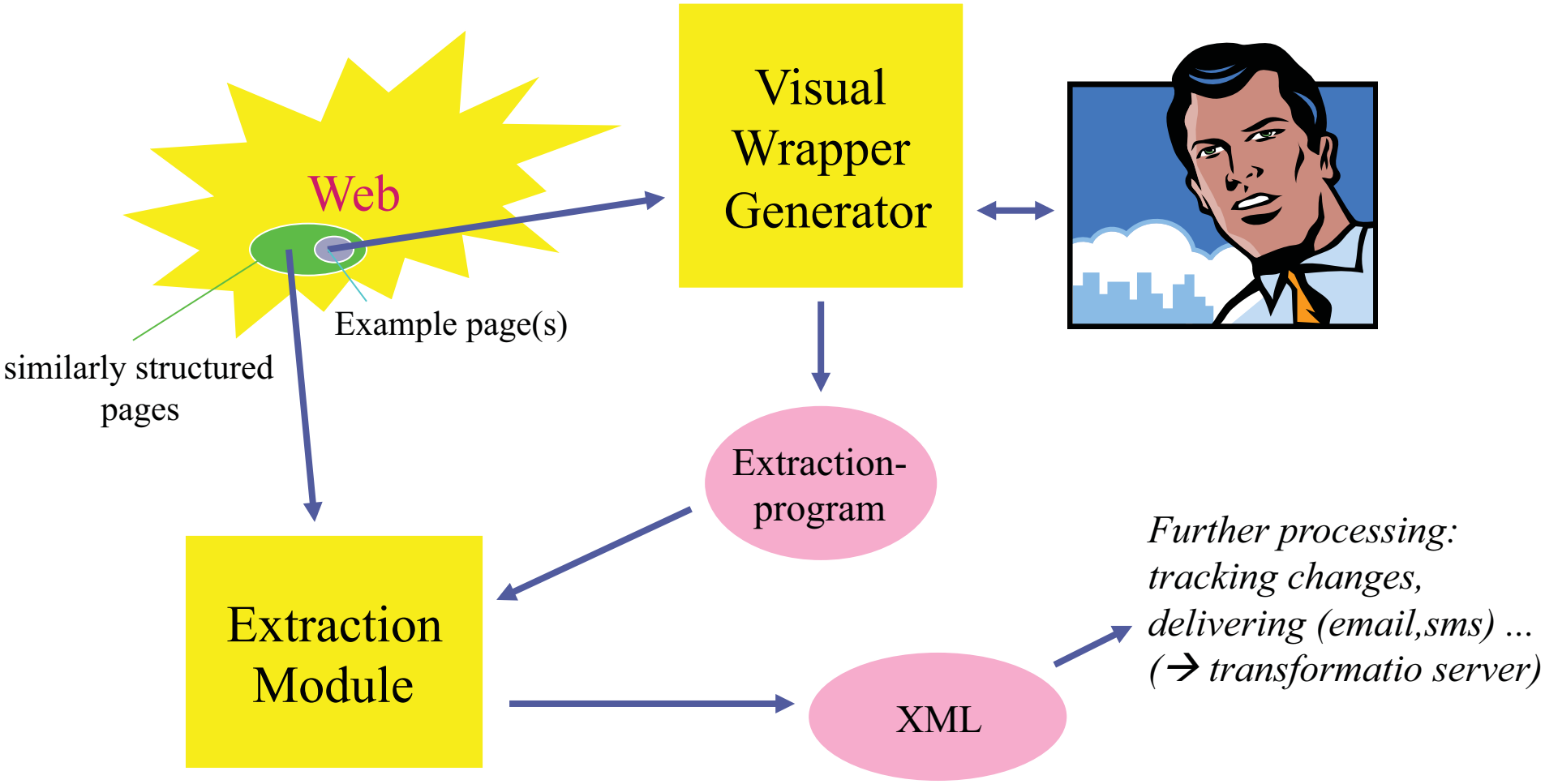
Elog

$\cap$

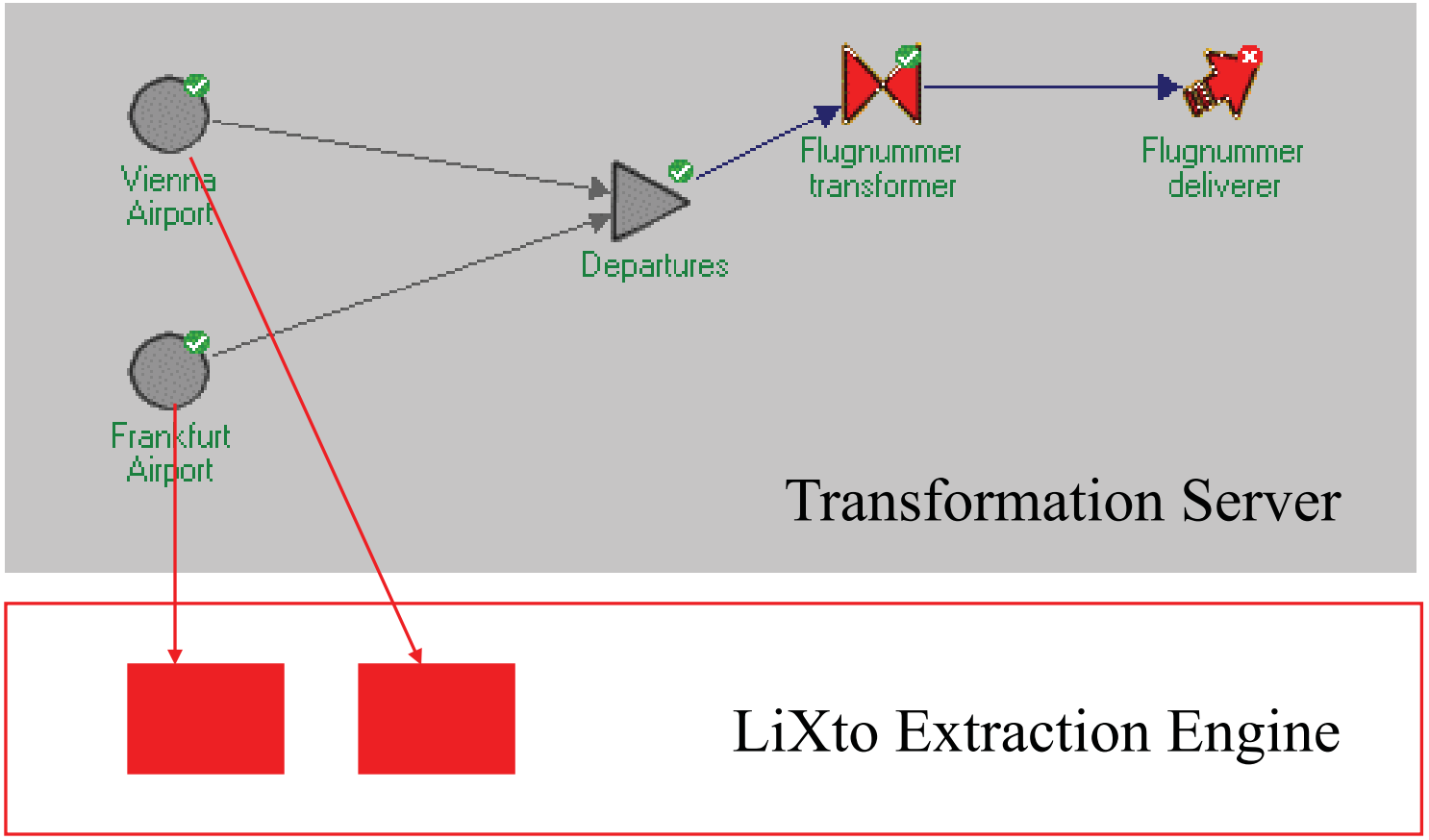
Lixto Visual Wrapper

(outil: suite logicielle)

Lixto Visual Wrapper Architecture



Product Architecture



SHORT DEMO

Talk Outline

- Motivation: need of information extraction
- Logical foundations of information extraction
- The Lixto Visual Wrapper
- The Diadem Project: Fully automatic data extraction

Need for Automatic Extraction Technology (2)

All search engine providers need it! Many work on it.

Keywords: → Vertical search,
→ object search,
→ semantic search.

Raghu Ramakrishnan, Yahoo!, March 2009:
"*no one really has done this successfully at scale yet*"

Alon Halevy, Google, Feb. 2009: "*Current technologies are not good enough yet to provide what search engines really need. […] any successful approach would probably need a combination of knowledge and learning.*"

ERC Advanced Grant
Research proposal (Part B1)

Domain-centric Intelligent Automated Data Extraction Methodology

DIADEM

Principal Investigator: Georg Gottlob
Host Institution: University of Oxford
Full title: Domain-centric Intelligent Automated Data Extraction Methodology
Short name: DIADEM
Duration in months: 60

Gottlob

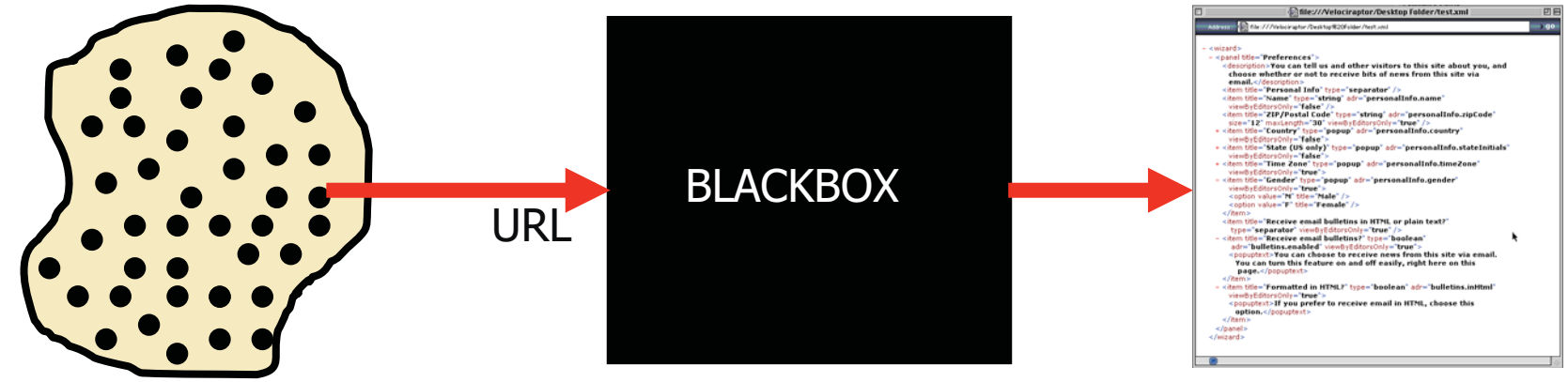
ERC Advanced Grant
Research proposal (Part B1)

Domain-centric Intelligent Automated Data Extraction Methodology

DIADEM

Principal Investigator: Georg Gottlob
Host Institution: University of Oxford
Full title: Domain-centric Intelligent Automated Data Extraction Methodology
Short name: DIADEM
Duration in months: 60

The Blackbox we are constructing



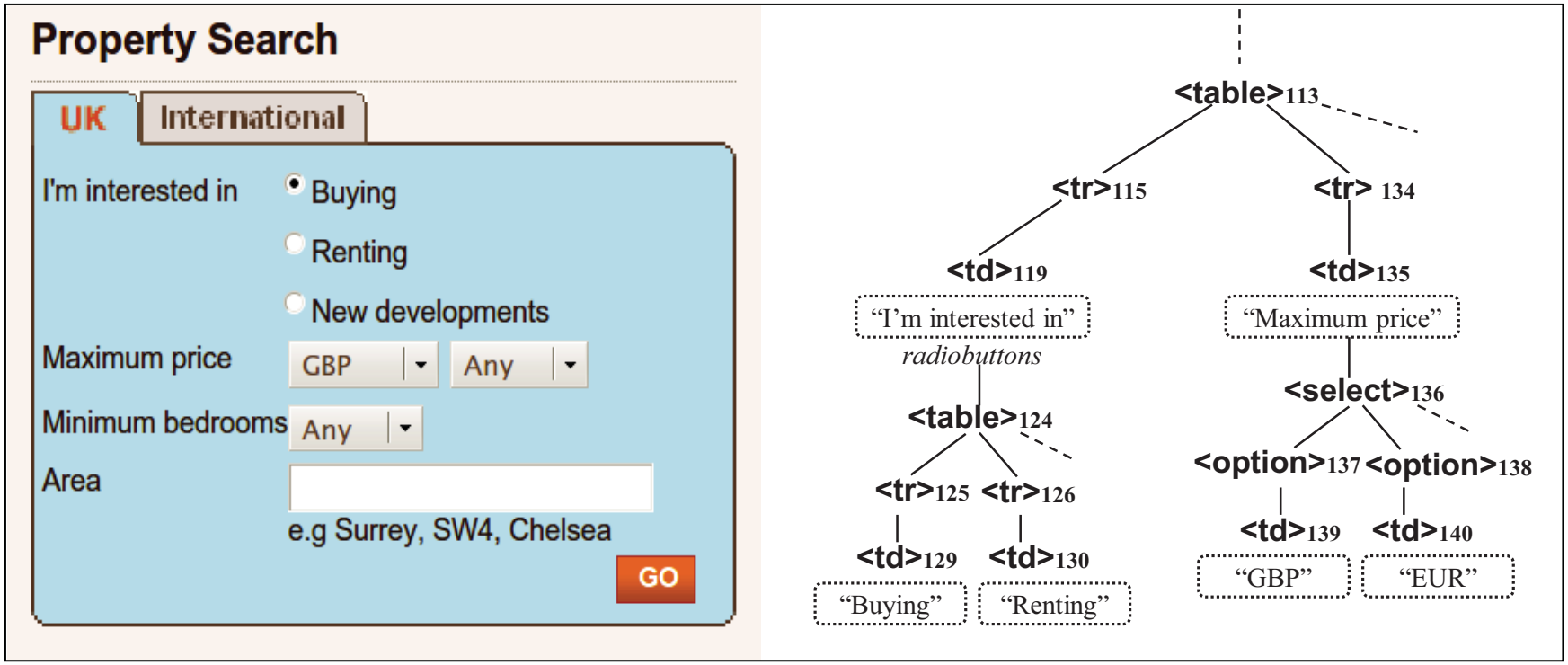
Application domain with thousands of websites

Application relevant Structured data (XML or RDF)

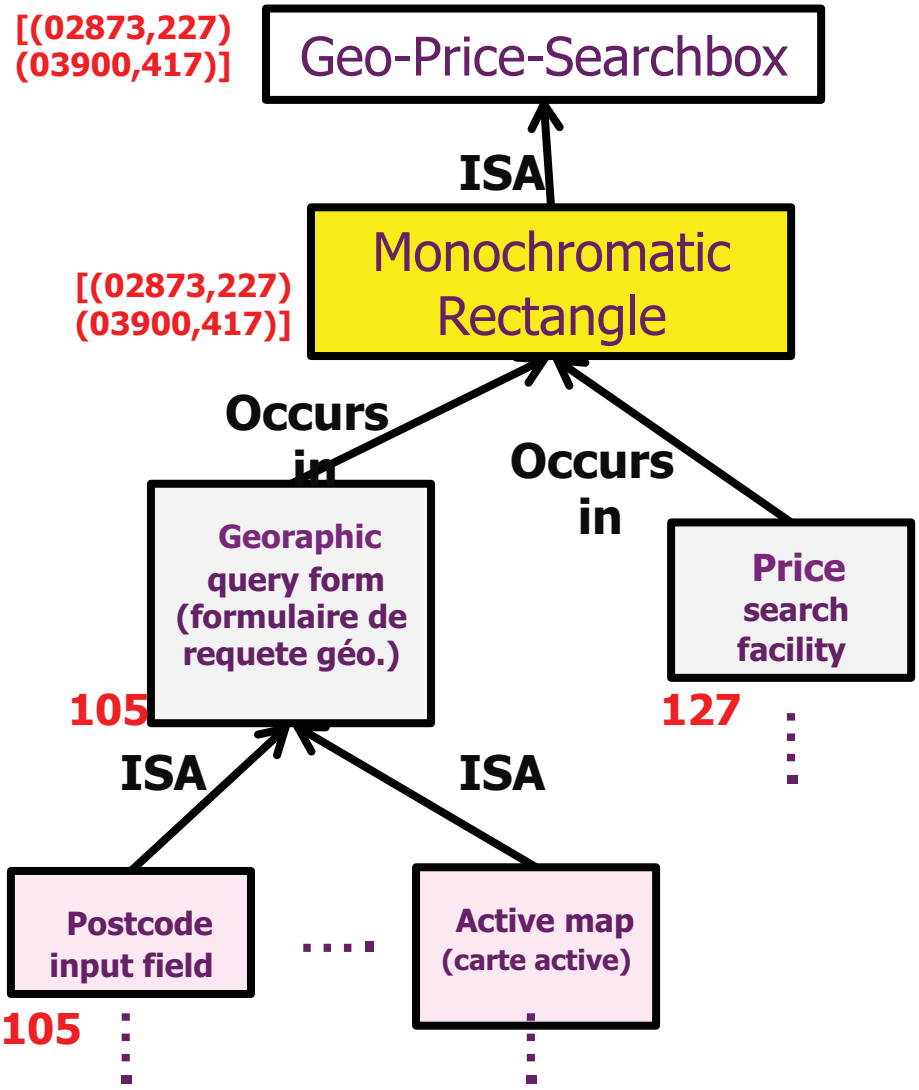
To achieve this, we combine a host of annotators with a new knowledge-based approach.

How to achieve it?

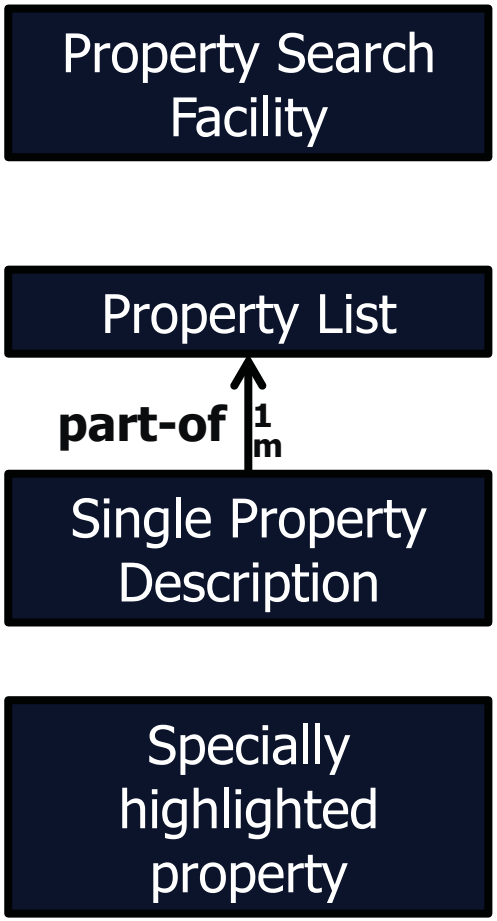
Combine existing and new “low level” annotators with “high level” AI and reasoning.


Property Search
UK
International
I'm interested in
Buying
Renting
New developments
Maximum price
GBP
Any
Minimum bedrooms
Any
Area
e.g Surrey, SW4, Chelsea
GO
<table>113
<tr>115
<tr> 134
<td>119
<td>135
"I'm interested in"
radiobuttons
"Maximum price"
<table>124
<select>136
<tr>125
<tr>126
<option>137
<option>138
<td>129
<td>130
<td>139
<td>140
"Buying"
"Renting"
"GBP"
"EUR"

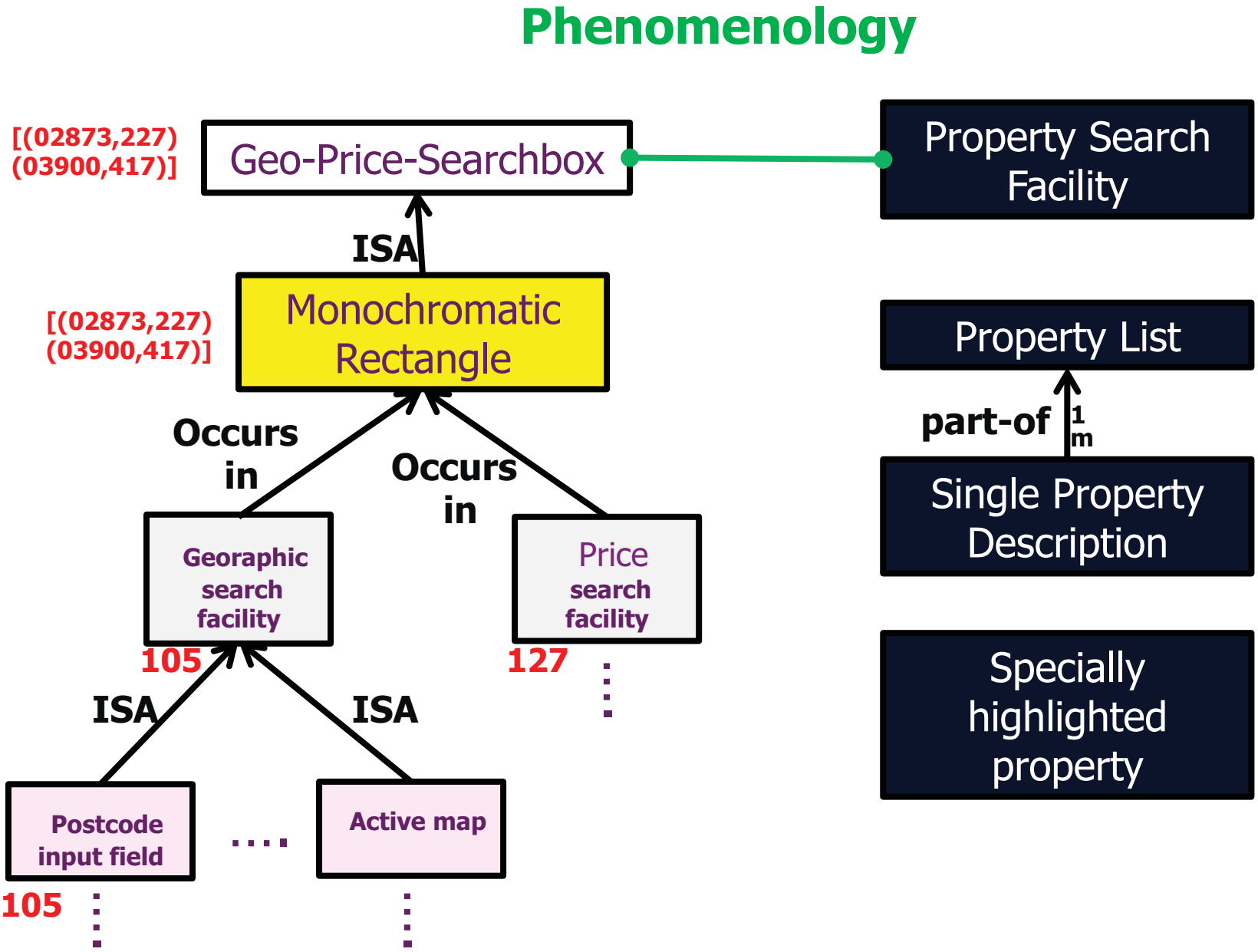
Bottom-up (low-level) annotation



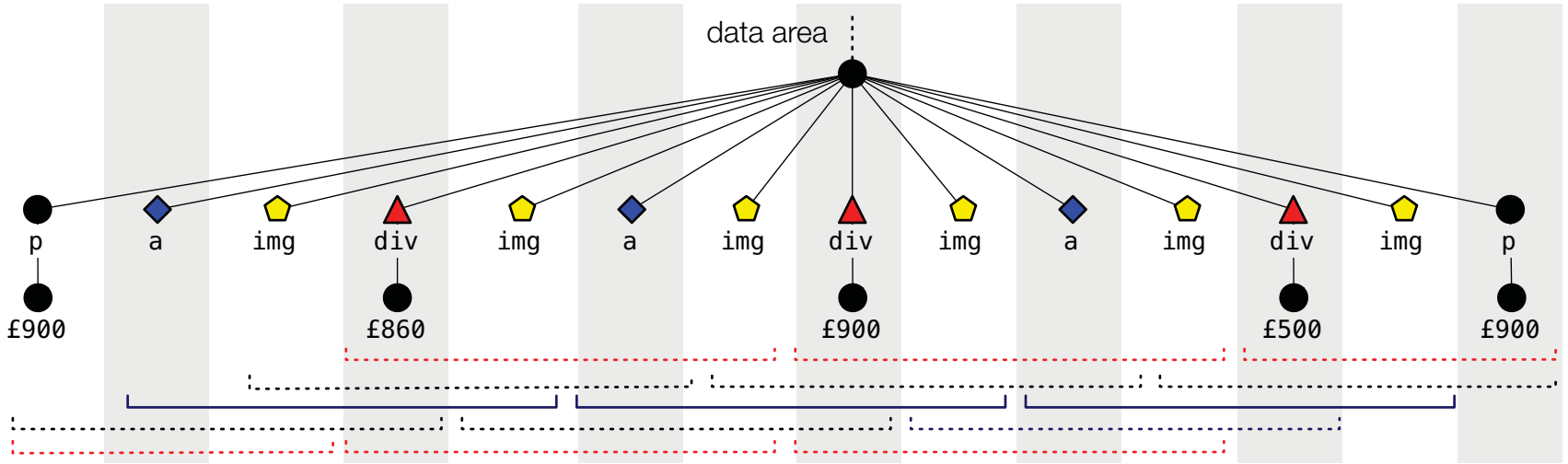
Top-down reasoning



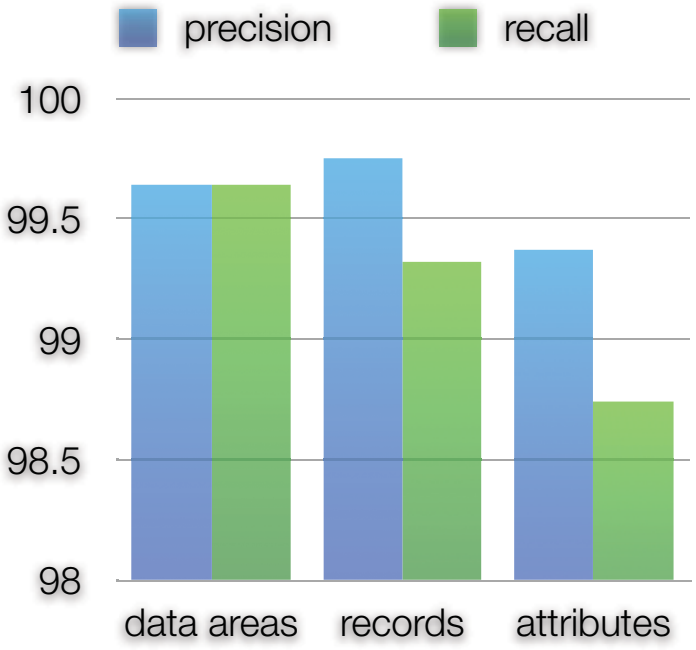
Bottom-up processing Top-down reasoning



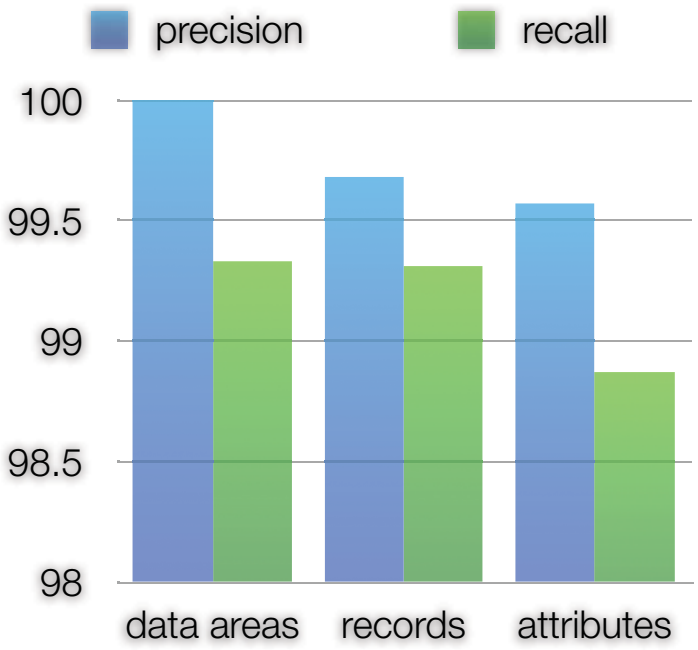
Phenomenological Record Segmentation



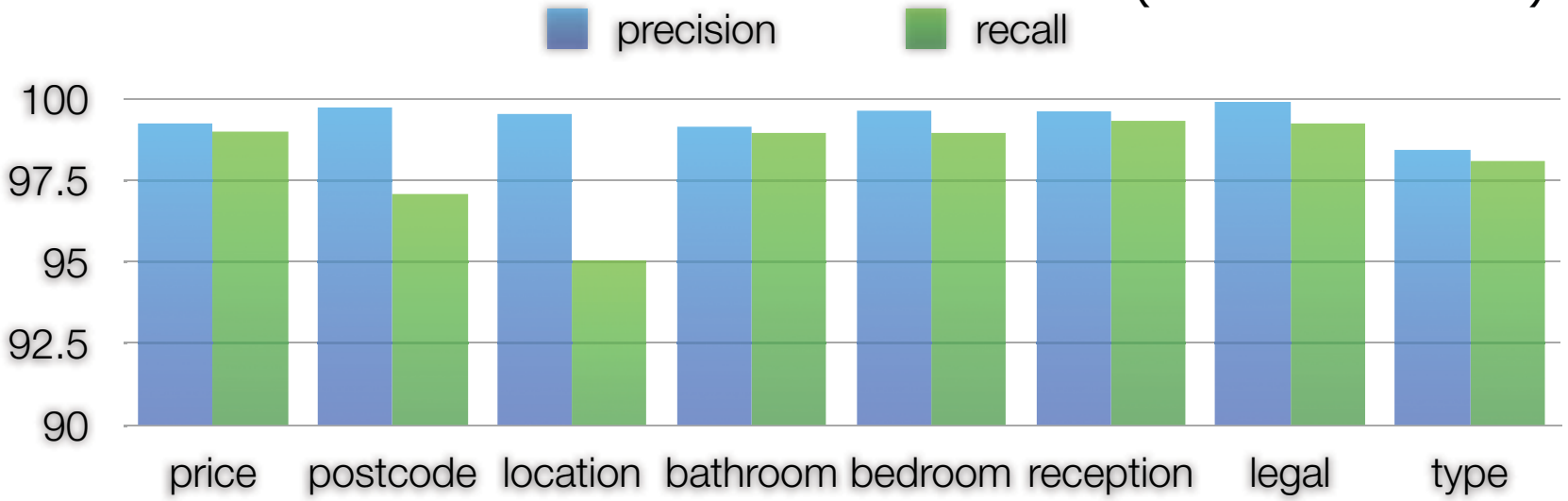
- set of uniform, non-overlapping records
- maximise sequence of evenly segmented (same distance pivot)
- minimise irregularity of records



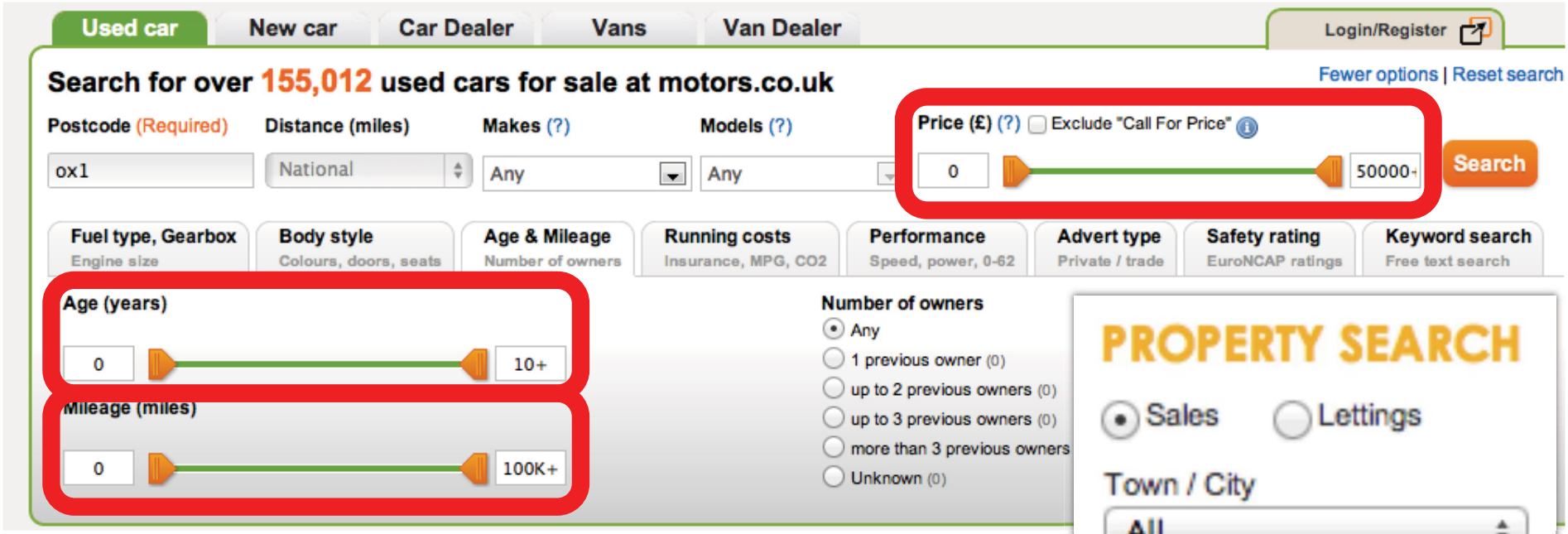
Real Estate
(100 pages)

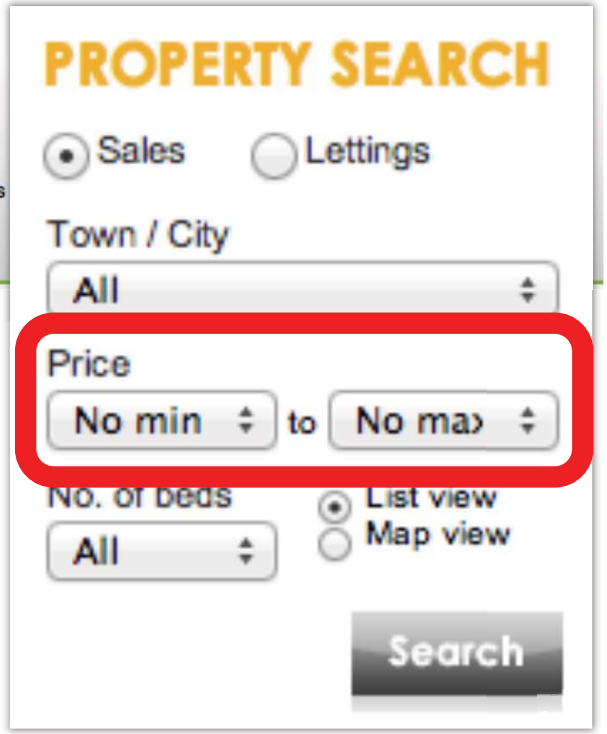


Used Car
(100 pages)
(voitures d'occasion)



Form Patterns Example

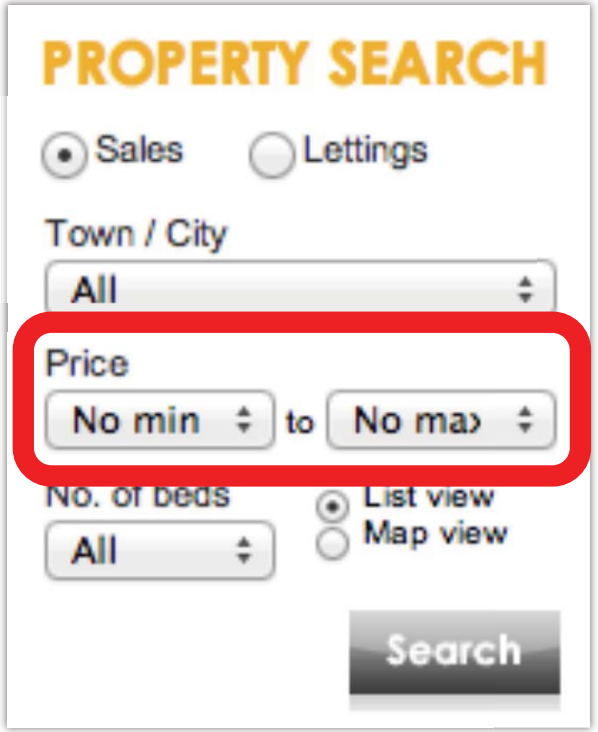




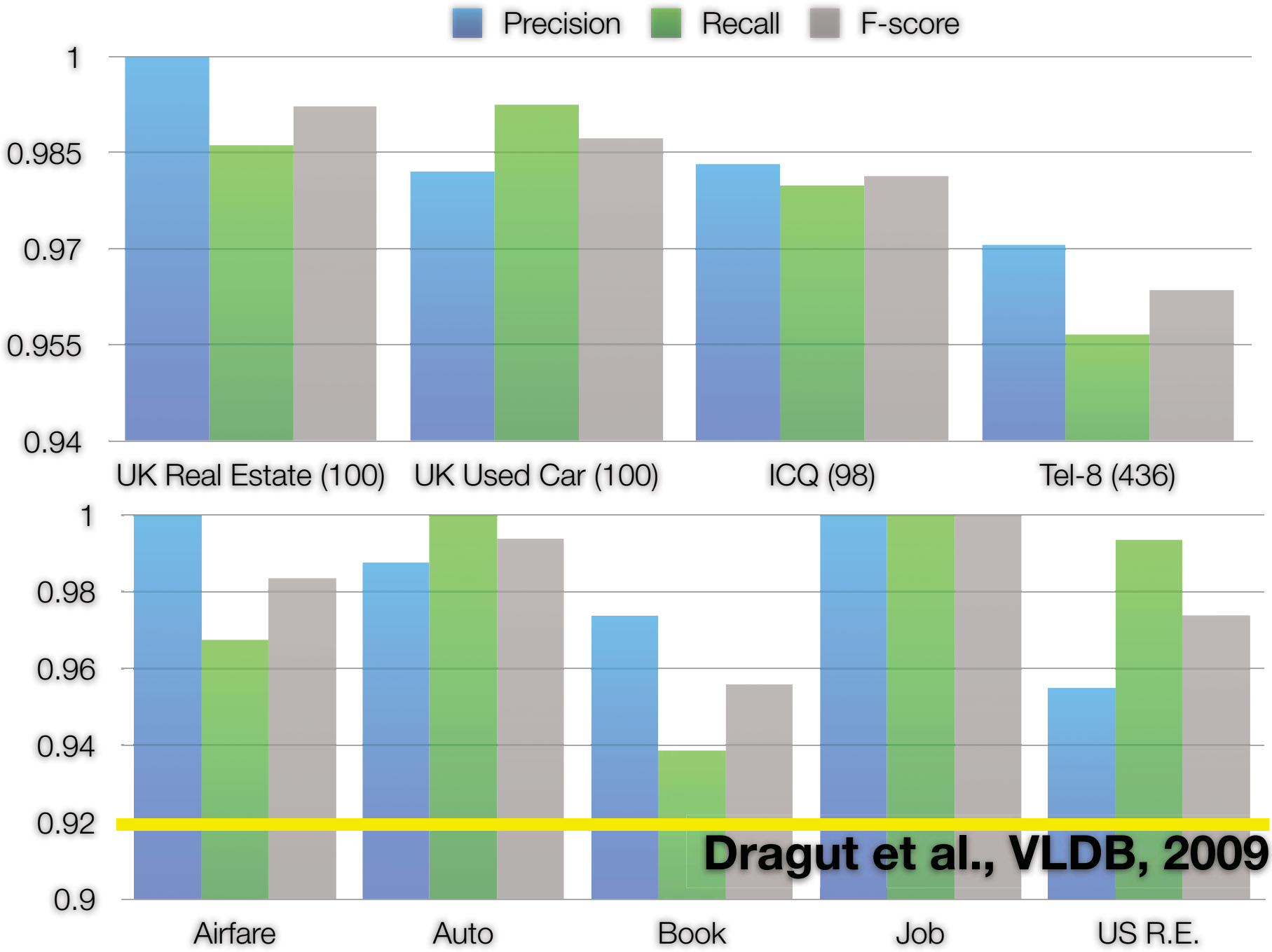
- Small set of ubiquitous patterns
  - ranges, dates, options, etc.
- Ontology by instantiation

OPAL-TL Example

- Price range
  - two successive fields in the same group
  - at least one “price” type
  - range connector in between



$$\texttt{concept<C}_M\texttt{>(N}_2\texttt{)} \Leftarrow \texttt{child(N}_1\texttt{,G),child(N}_2\texttt{,G),follows(N}_2\texttt{,N}_1\texttt{),}$$
$$\texttt{concept<C>(N}_1\texttt{),N}_2\texttt{@}\mathit{range\_connector}\texttt{\{e,d\},}\neg\texttt{(A}_1 \prec A\texttt{, N}_2\texttt{@A}_1\texttt{\{d\})}$$


Precision
Recall
F-score
1
0.985
0.97
0.955
0.94
UK Real Estate (100)
UK Used Car (100)
ICQ (98)
Tel-8 (436)
1
0.98
0.96
0.94
0.92
0.9
Airfare
Auto
Book
Job
US R.E.
Dragut et al., VLDB, 2009

Short Demo diadem-3min43.m4v