Data Querying, Extraction and Integration II: Applications

Recuperación de Información
2007
Lecture 5.

Goal today: Provide examples for useful XML based applications

- Motivation:
  - Integrating Legacy Databases, etc.
  - Extractions from Websites
- Wrapper Generators for the Web
- APIs, Software
- Further interesting XML standards and applications
- What's missing?

Problems with the Data Integration: extracting information

- Different formats, different syntax, etc.
- few proprietary standards, e.g. EDIFACT, SWIFT, etc.

Fahrradhandel Pedal, Wagingerstr. 5, 81549 München

Huber GmbH
Obstgasse 2
81549 München

München, 02.08.99

Rechnung:
9908001

Ihre Bestellung Nr. O0010001 vom 15.07.99

Pos	Artikel	Beschreibung	Anzahl	Einzelpreis	Gesamt
1	4711.001	Fahrrad, Damen-	1	750,00	750,00
2	4711.002	Luftpumpe, Stand-	1	19,90	19,90
3	4711.003	Ersatzventil	3	2,50	7,50
		Gesamtsumme netto			777,40
		Umsatzsteuer 16%			124,38
		zu zahlender Betrag			901,78

Alle Beträge verstehen sich in DEM

```
UNA:+,? '
UNB+UNOA:2+FHPEDAL+HUBERGMBH+9908
02:1557+9908021557'
UNH+INVOIC0001+INVOIC:D:93A:UN'
BGM+380+9908001+9' DTM+3+19990802+102'
RFF+ON+O0010001' DTM+4+19999715:102'
NAD+SE++Fahrradhandel Pedal++Wagingerstr.
5+München++81549' NAD+BY++Huber
GmbH++Obstgasse 2+München++81549'
LIN+1++4711.001' IMD+F++:::Fahrrad, Damen'
QTY+47:1:PCE' MOA+66:750' PRI+AAA:750'
LIN+2++4711.002' IMD+F++:::Luftpumpe, Stand-'
QTY+47:1:PCE' MOA+66:19,9' PRI+AAA:19,9'
LIN+3++4711.003' IMD+F++:::Ersatzventil'
QTY+47:3:PCE' MOA+66:7,5' PRI+AAA:2,5'
UNS+S' MOA+79:777,4' MOA+124:124,38'
MOA+128:901,78' TAX+7+VAT+++:::16+S'
UNT+28+INVOIC0001' UNZ+1+9908021557'
```

Fig. 4 bill as usual (left) and the EDIFACT message (right)

Problems with the Data Integration: combining information

Fahrradhandel Pedal, Wagingerstr. 5, 81549 München

Huber GmbH
Obstgasse 2
81549 München

München, 02.08.99

Rechnung:
9908001

Ihre Bestellung Nr. O0010001 vom 15.07.99

Pos	Artikel	Beschreibung	Anzahl	Einzelpr
1	4711.001	Fahrrad, Damen-	1	750
2	4711.002	Luftpumpe, Stand-	1	19
3	4711.003	Ersatzventil	3	2

Gesamtsumme netto
Umsatzsteuer 16%

zu zahlender Betrag

Alle Beträge verstehen sich in DEM

Fig. 4 bill as usual (

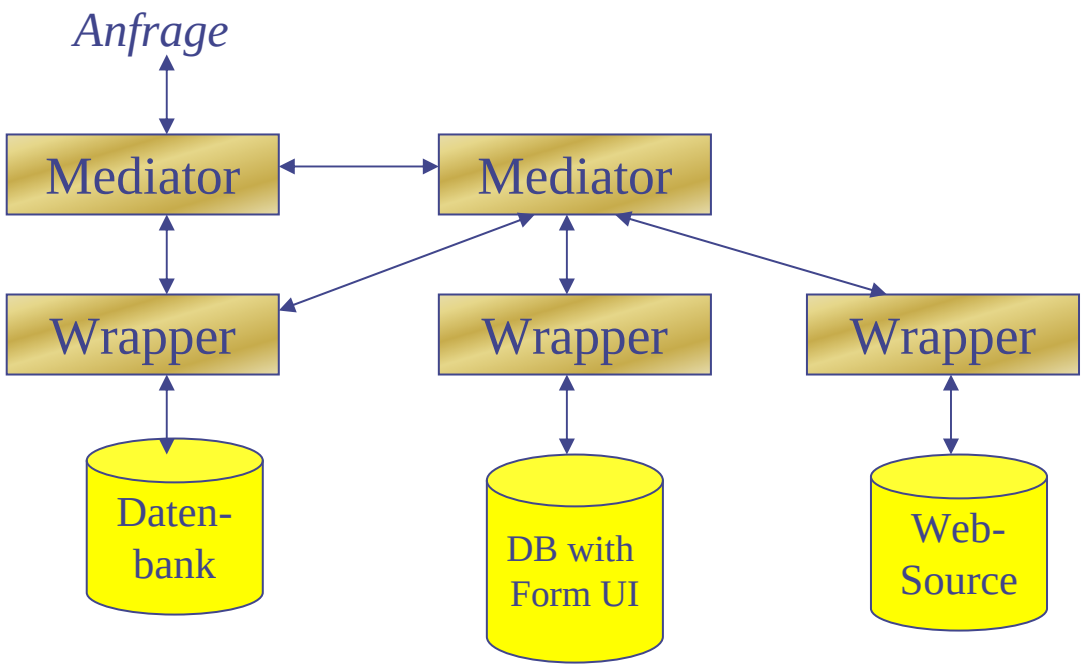
```
UNA:+,? '
UNB+UNOA:2+FHPEDAL+HUBERGMBH+9908
02:1557+9908021557'
UNH+INVOIC0001+INVOIC:D:93A:UN'
BGM+380+9908001+9' DTM+3+19990802+102'
RFF+ON+O0010001' DTM+4+19999715:102'
NAD+SE++Fahrradhandel Pedal++Wagingerstr.
5+München++81549' NAD+BY++Huber
```

```
{1:F01MIDLGB22AXXX0548034693}{2:I103BKTRUS33XBRDN3}{3:{108:MT103}}{4:
:20:8861198-0706
:23B:CRED
:32A:000612USD5443,99
:33B:USD5443,99
:50K:GIAN ANGELO IMPORTS
NAPLES
:52A:BCITITMM500
:53A:BCITUS33
:54A:IRVTUS3N
:57A:BNPAFRPPGRE
:59:/20041010050500001M02606
KILLY S.A.
GRENOBLE
:70:/RFB/INVOICE 559661
:71A:SHA
-}
```

Possible Solutions:

- Write mediators from scratch, or:

- Use Wrappers and Mediators!
- Common format (XML)
- Easy transformation (XSLT)



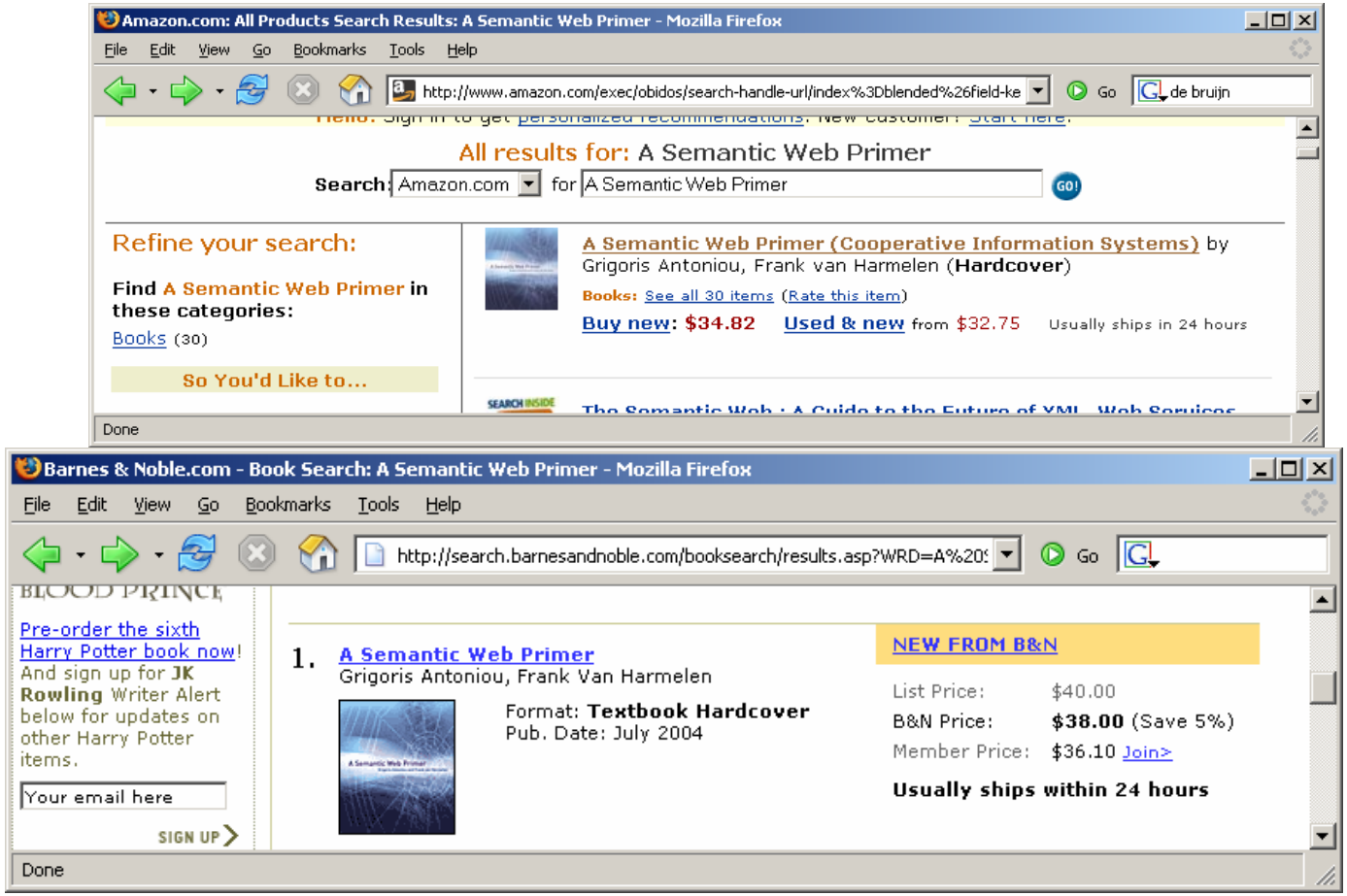
Analogous problems with the Web: extracting information

Which book is about the Web?

What is the price of the book?

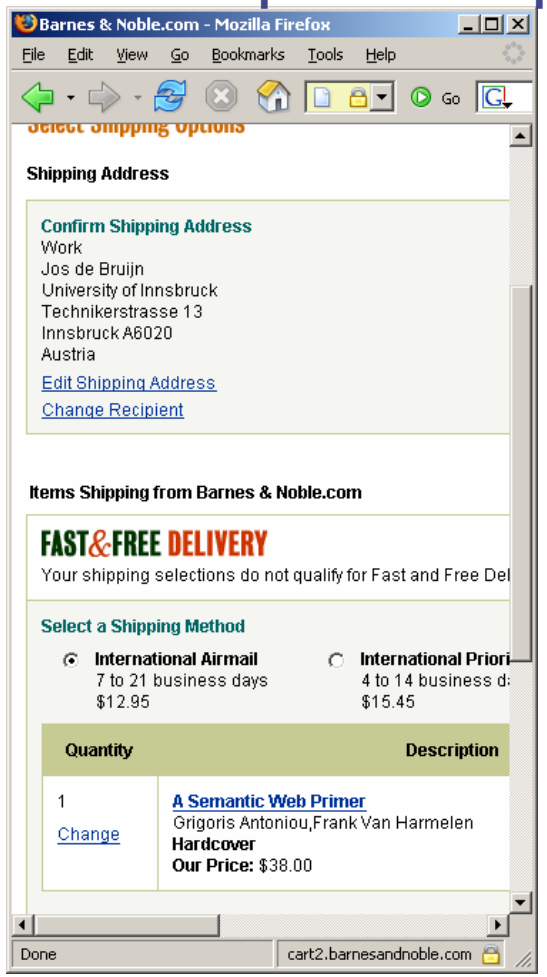
Problems with the Web: combining information

I want the cheapest copy of "A Semantic Web Primer".



The Problems with the Web: combining information

I want the cheapest copy of "A Semantic Web Primer"; taking into account the price for shipping the book.



On average 10 clicks to find out what the shipping rate is!

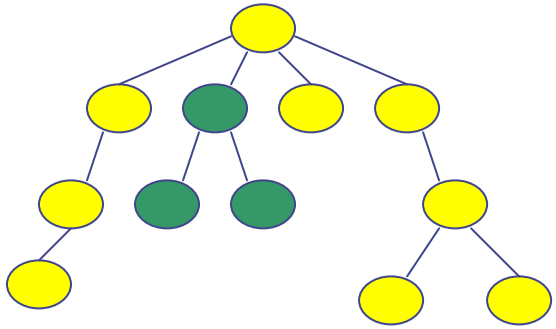
The solution particularly for Web integration:

- 2 alternatives:
  - **Top-down: Create wrappers for current web sites and extract data automatically (wrappers)**
    - ***Today we mainly focus on this part!***
  - Bottom-up: Instead of publishing natural language, publish machine-processable data directly (semantic Web idea!)!

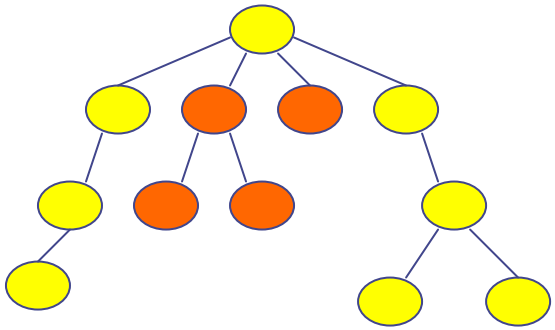
Wrappers for Websites:

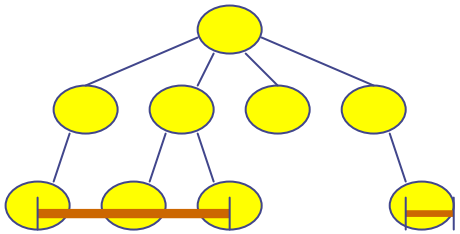
- Create XML from Websites
- Advantage for Web-based integration: HTML from websites is already very close to XML (or even XTML already!)
- with XSLT, XPath, etc. we already have almost everything done (for tree-based wrapping)!

Wrappers for Websites – Ways of extraction:



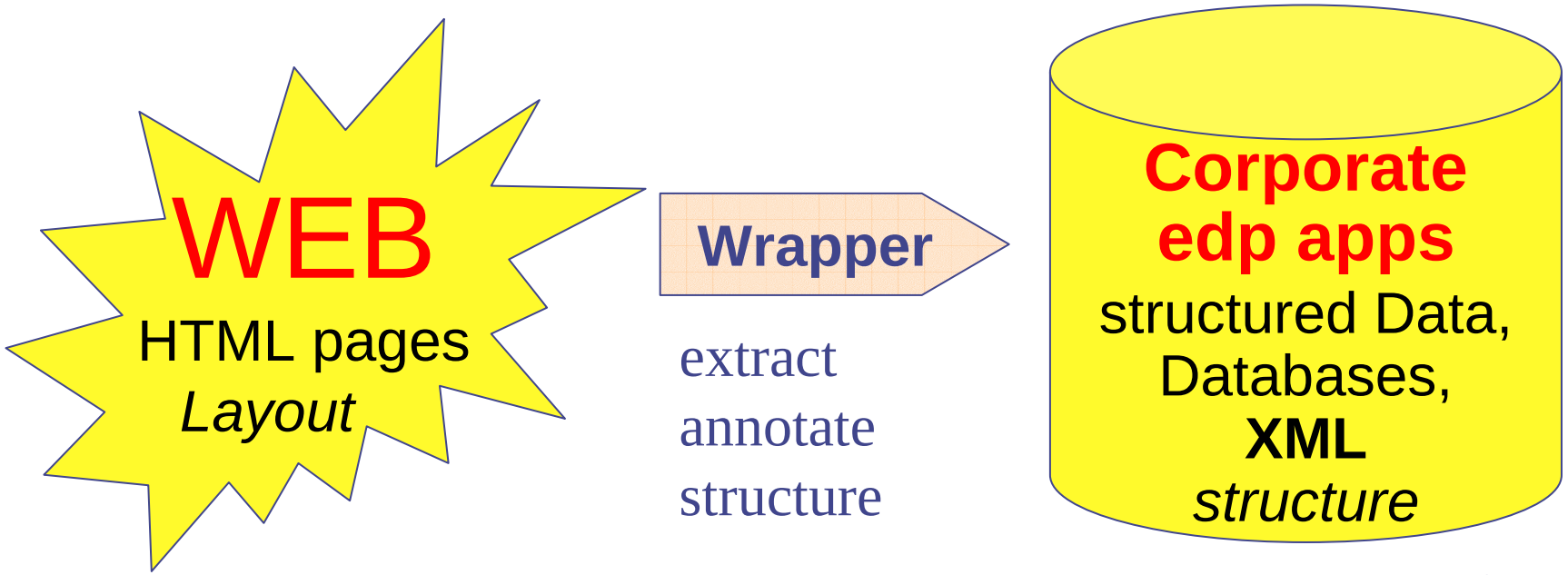
- Tree extraction
  - Distinction "subtree" und "tree region"
  - A subtree is represented by ist root (a node in the original tree)
  - A tree region consists of a list of sibling nodes.
- Stringextraktion
  - Operates on substrings of the (HTML) Document, e.g. by regular expressions.



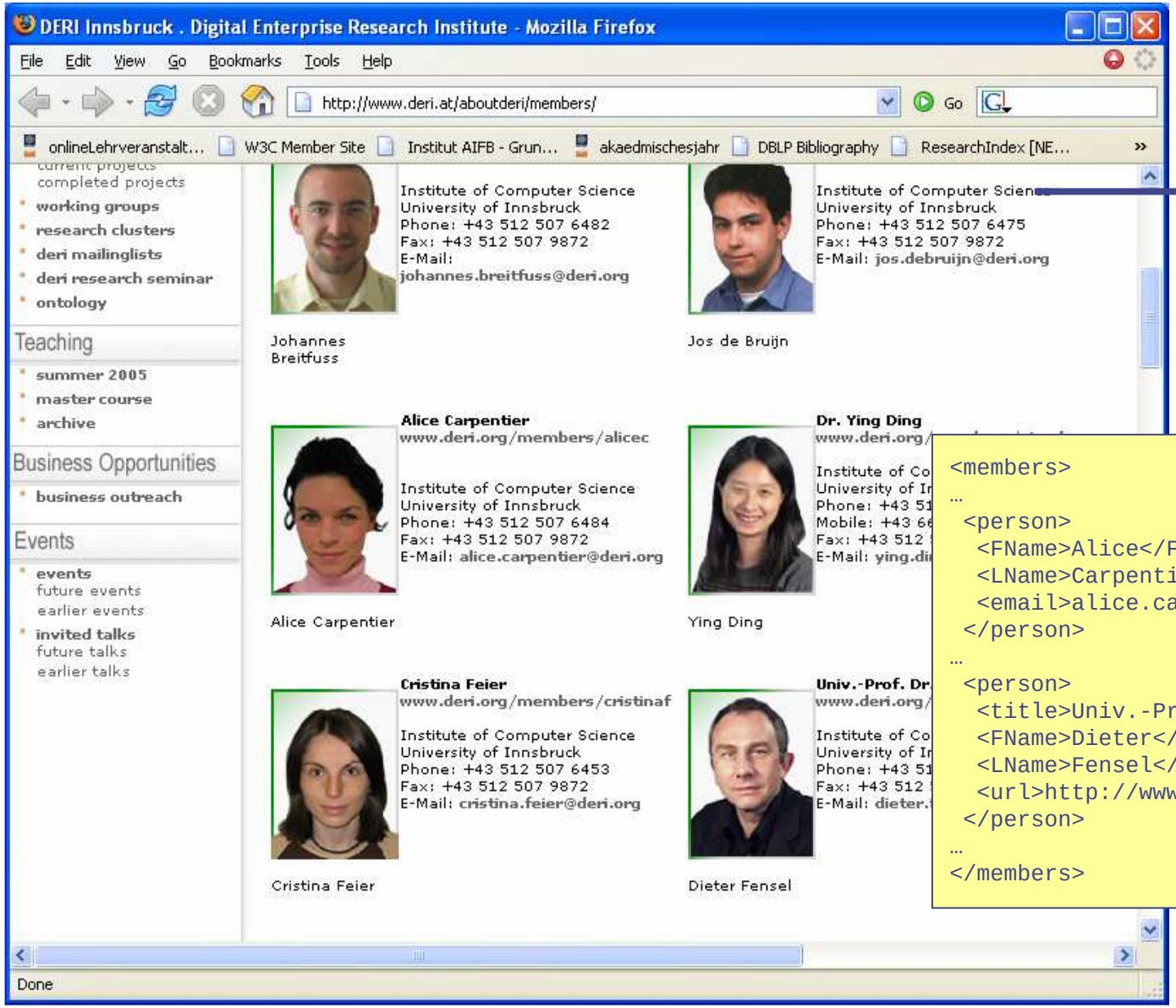


Motivation for Web Extraction: Bridge the Gap

Goal: Make Web content accessible for electronic data exchange.



An Example: deri.at members page




```
<members>
…
 <person>
  <FName>Alice</FName>
  <LName>Carpentier</LName>
  <email>alice.carpentier@uibk.ac.at</email>
 </person>
…
 <person>
  <title>Univ.-Prof. Dr. </title>
  <FName>Dieter</FName>
  <LName>Fensel</LName>
  <url>http://www.fensel.com</url>
 </person>
…
</members>
```

Other examples:

- Combine results from different search engines
- Stock quotes
- News filters (from several pages)
- Price-comparison (e.g. I want to by a laptop)
- Keep informed about concurrent competitors
- etc.

Common Task:

Similar looking pages, where you want to semi-automatically wrap information out of the HTML structure.

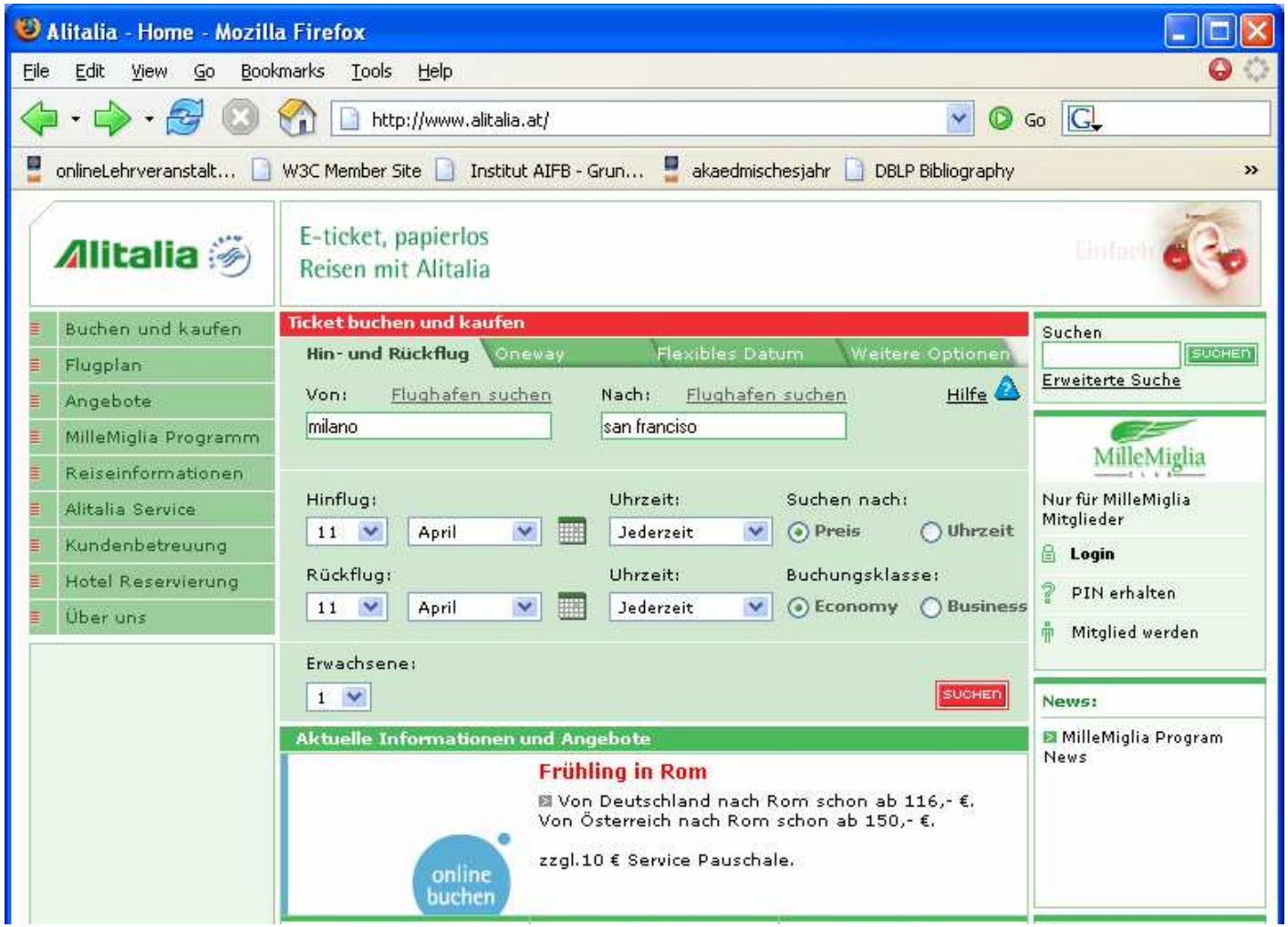
Additional, useful features would be:

- Automatic requerying schedule
- Automatization of forms and login/access control!
- notification of the wrapper does not work any longer due to site changes, etc.

Extraction from Websites

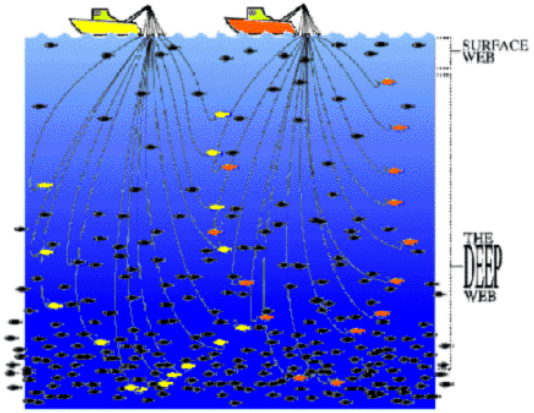
- more problems before actual extraction step:

  Most services on the web interact with Web interfaces (forms, applets, htaccess, logins, etc.)…

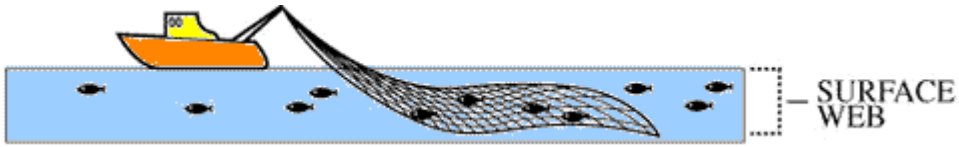


Extraction from Websites

- … the real data is hidden in this "deep web"



- much more data than on the "publicly indexable" surface web



- "The invisible portion of the Web will continue to grow exponentially before the tools to uncover the hidden Web are ready for general use"
  http://www.press.umich.edu/jep/07-01/bergman.html

- Need for **scripting** of interaction with forms, etc. for dynamic wrappers!

Web integration and extraction – Basics 1/3
HTTP Protocol

- Web Servers use HTTP Protocol:
- Request: HTTP GET, HTTP POST:

Request method GET or POST

Requested source

```
GET /main.html HTTP/1.0
accept:   image/gif,   image/x-xbitmap,   image/jpeg,   image/pjpeg,
application/vnd.ms-powerpoint,            application/vnd.ms-excel,
application/msword, */*
proxy-connection: Keep-Alive
host: www.philips.de
accept-language: de
connection: keep-alive
user-agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 4.0)

<request body>
```

host

Header can additionally contain info on e.g. whether cookies are accepted, etc.

Normally empty for HTTP GET, parameter data for HTTP POST

Example of how parameters can be encoded in a get request:

GET /search/de?o=1&p=chocolate&o=i&h=c&g=0&n=20 HTTP/1.0

…

host: de.search.yahoo.com

Web integration and extraction – Basics 1/3
HTTP Protocol

- HTTP Reply:

Some alternatives: 404 (not found), 302 (page moved)

```
HTTP/1.0 200 OK
proxy-connection: keep-alive
accept-ranges: bytes
cache-control: no-cache
content-type: text/html
connection: keep-alive
content-length: 2627
server: Apache/1.3.12 (Unix) (Red Hat/Linux) PHP/4.0.6
mod_perl/1.21

<HTML>…
```

Content type!

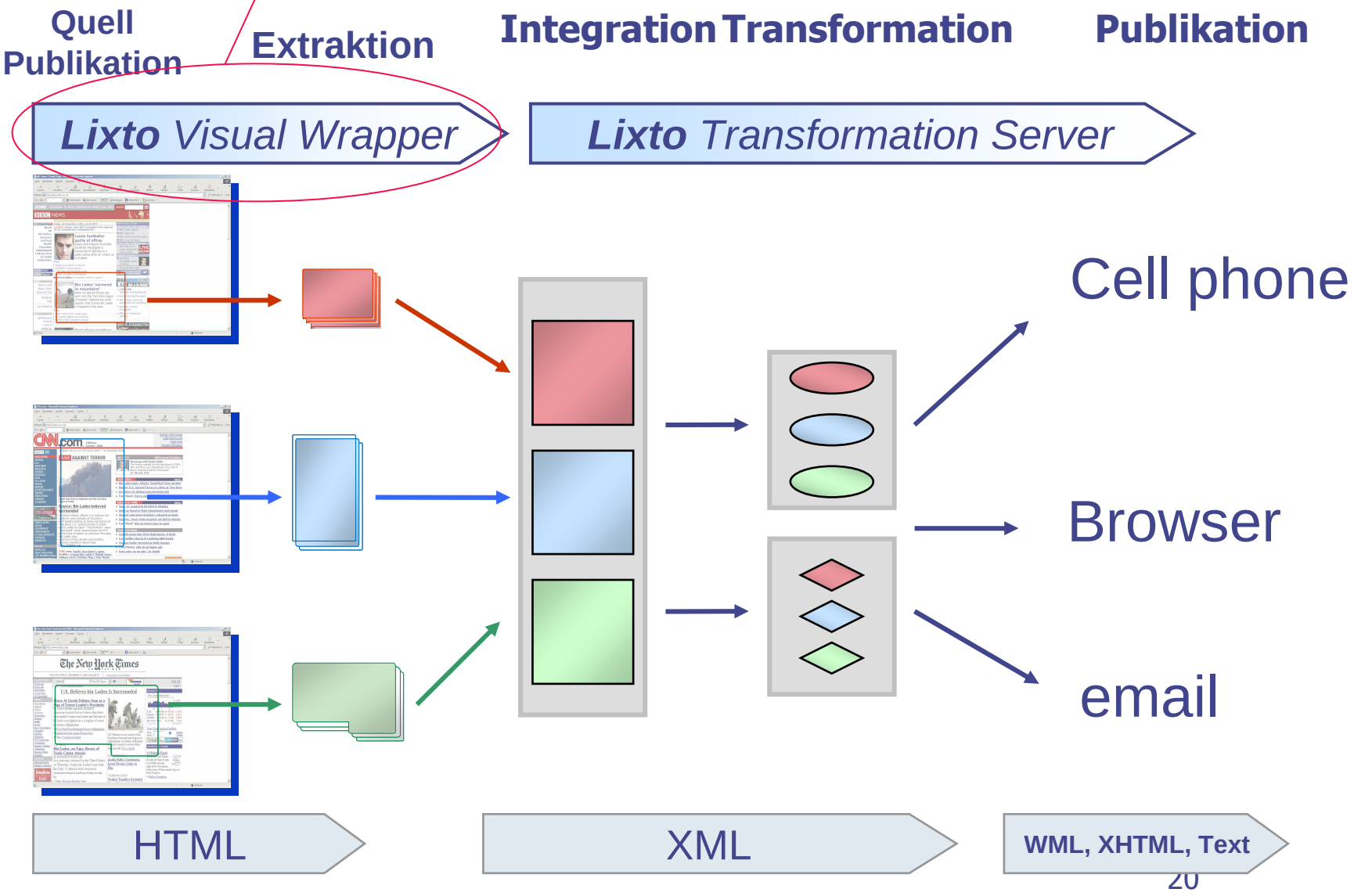
Usually HTML

We need to know these basics in order to write some automatic wrappers…

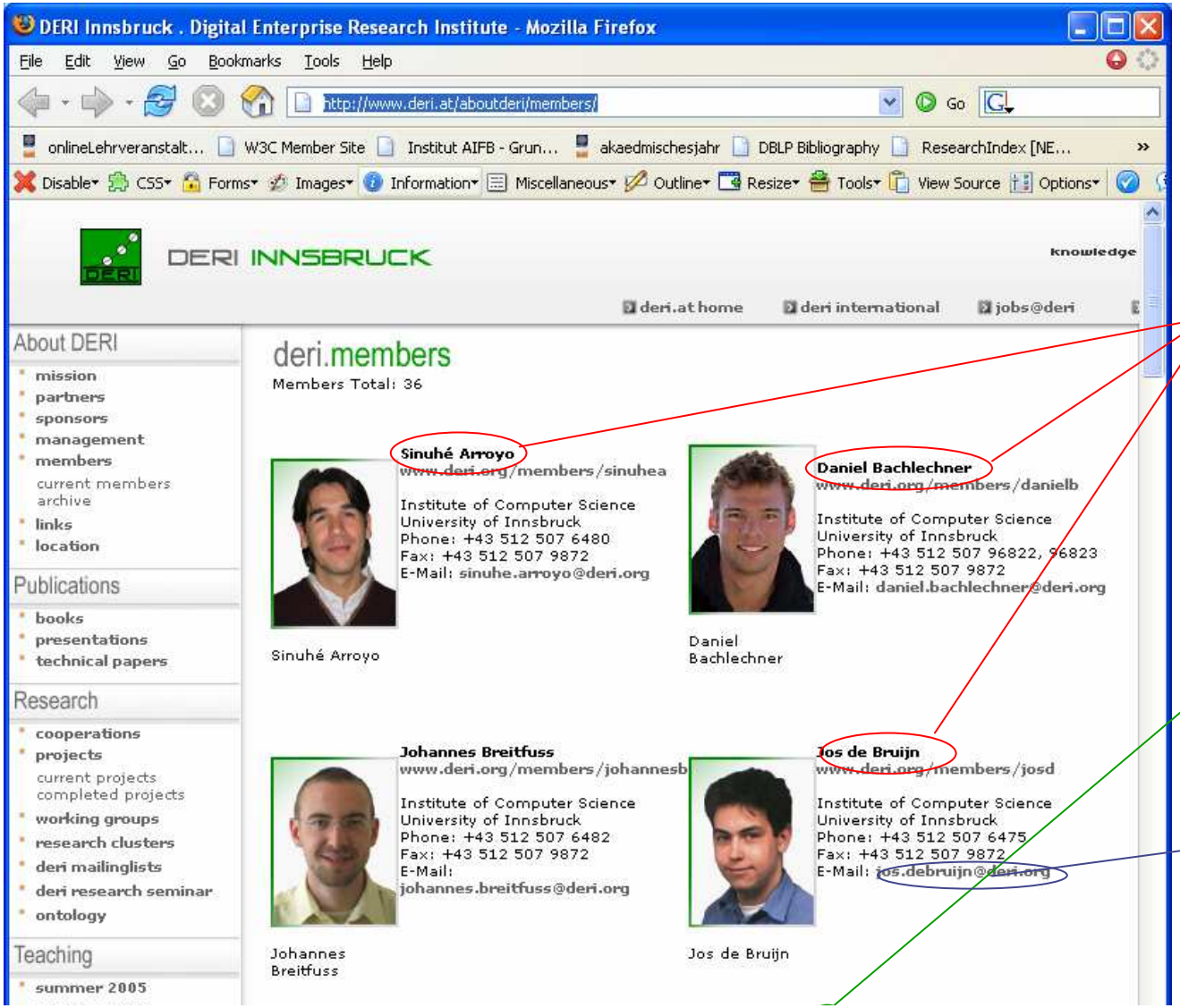
Website Wrapper Tools examples

- A compex example: LiXto Visual Wrapper (elog): not only for Web sources, also PDF, etc.
- Back to basic! Tidy+XSLT transformations

We'll only briefly demonstrate the visual wrapper here



Our example:



Patterns:

Name: bold, always on a similar positionIn the table

Title: always occurs excactly before name as part of the name

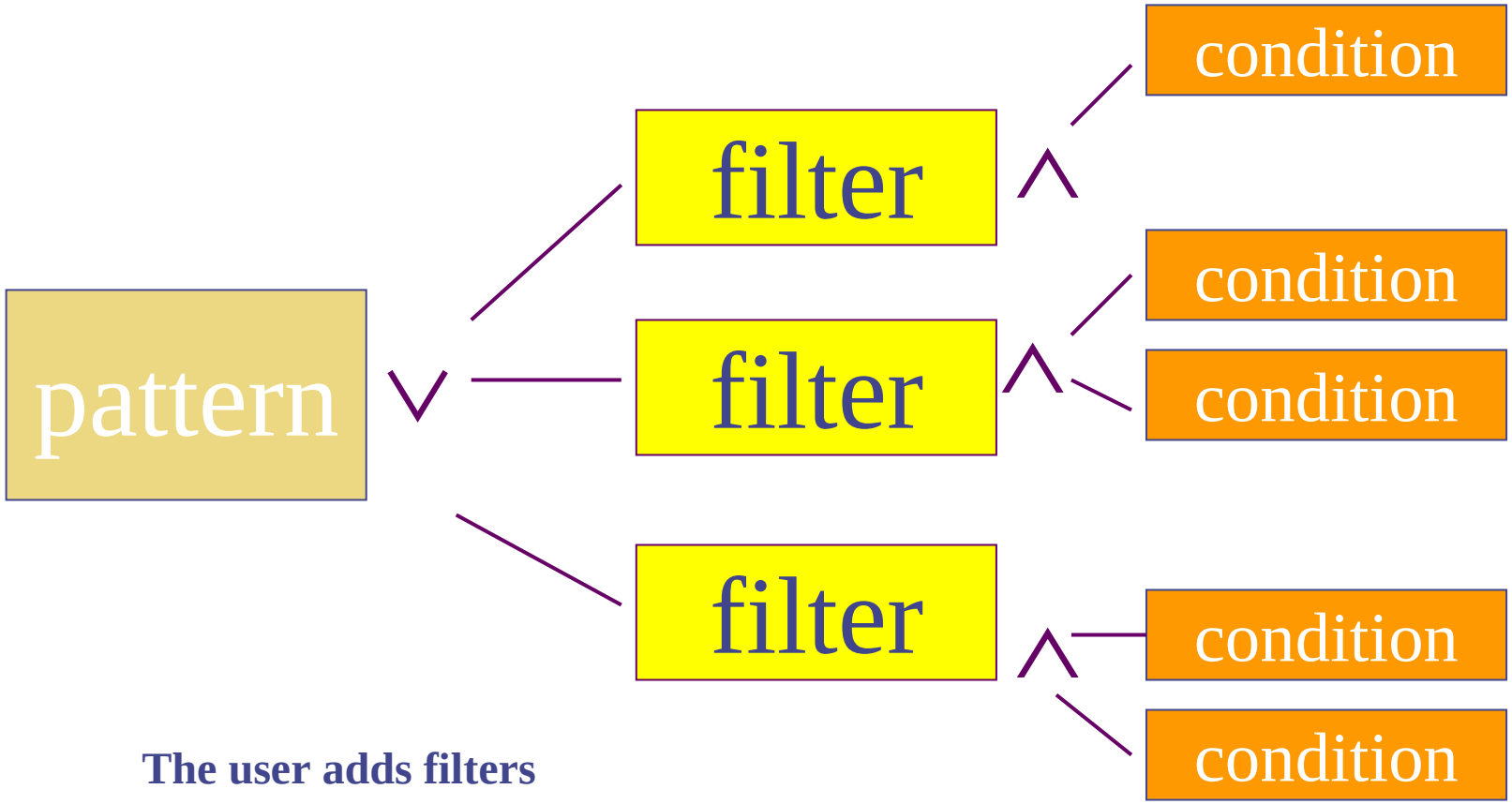
Email: is a link, preceeded by the String "E-Mail:"

Lixto: Wrapper Generation

Different approach to XSLT:

- **Example-based** approach
- **Visual** interaction
- Decalarative Language Elog
- Hierachical Extraction using Patterns & Filters.
- Ectract XML with XML Tool
- Tree-based and string-based, regular expressions
- "Fuzzy" filter functions, etc.

Patterns in Lixto:



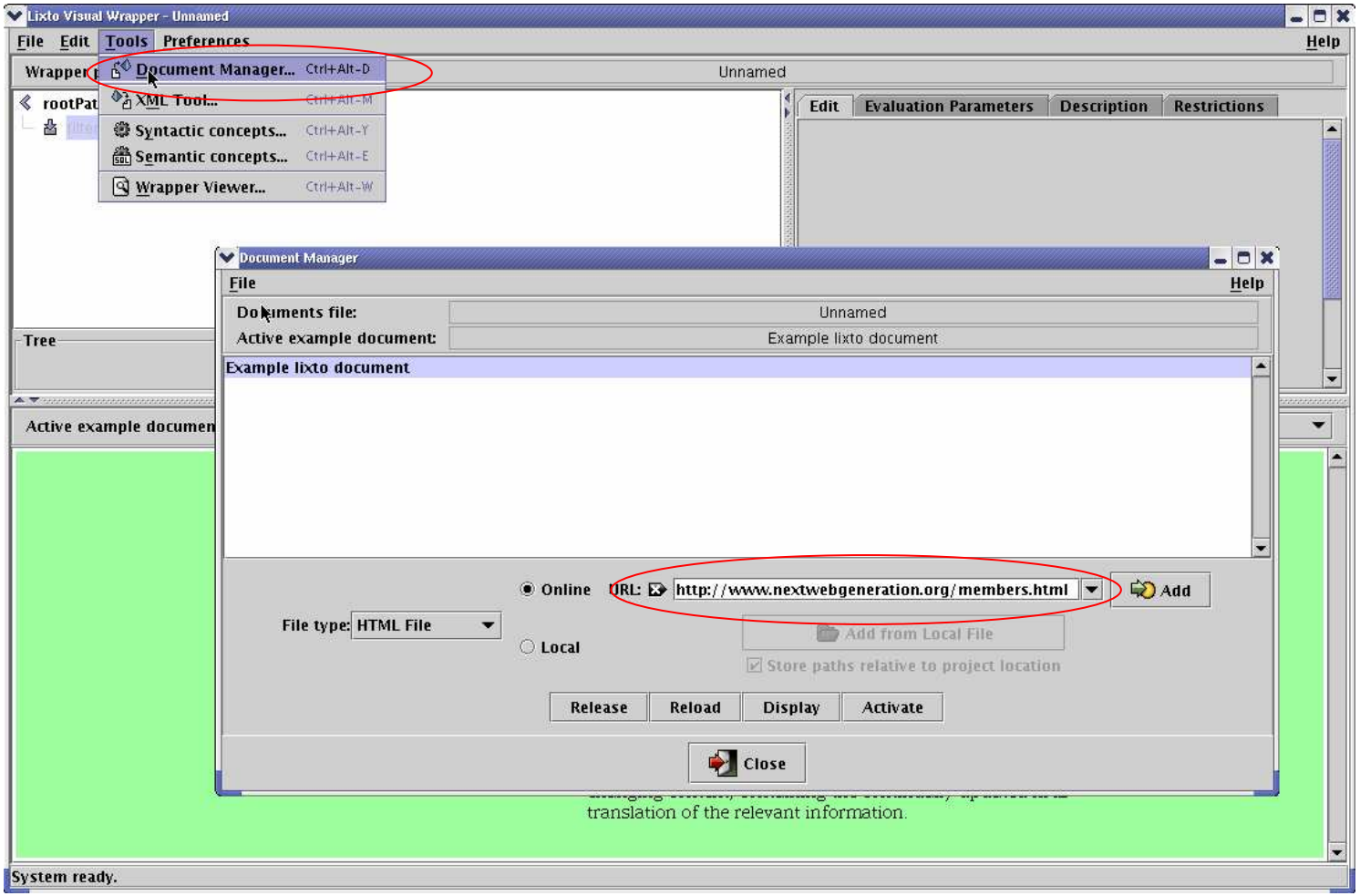
The user adds filters and conditions, characterizing the wanted information patterns by stepwise refinement/extension

- A Wrapper in Lixto consists of a set of (hierarchical) patterns
- Each filter extracs a set of instances
- Conditions restrict the numbers of instances
- The instances of a pattern is the union of the instances of its filters

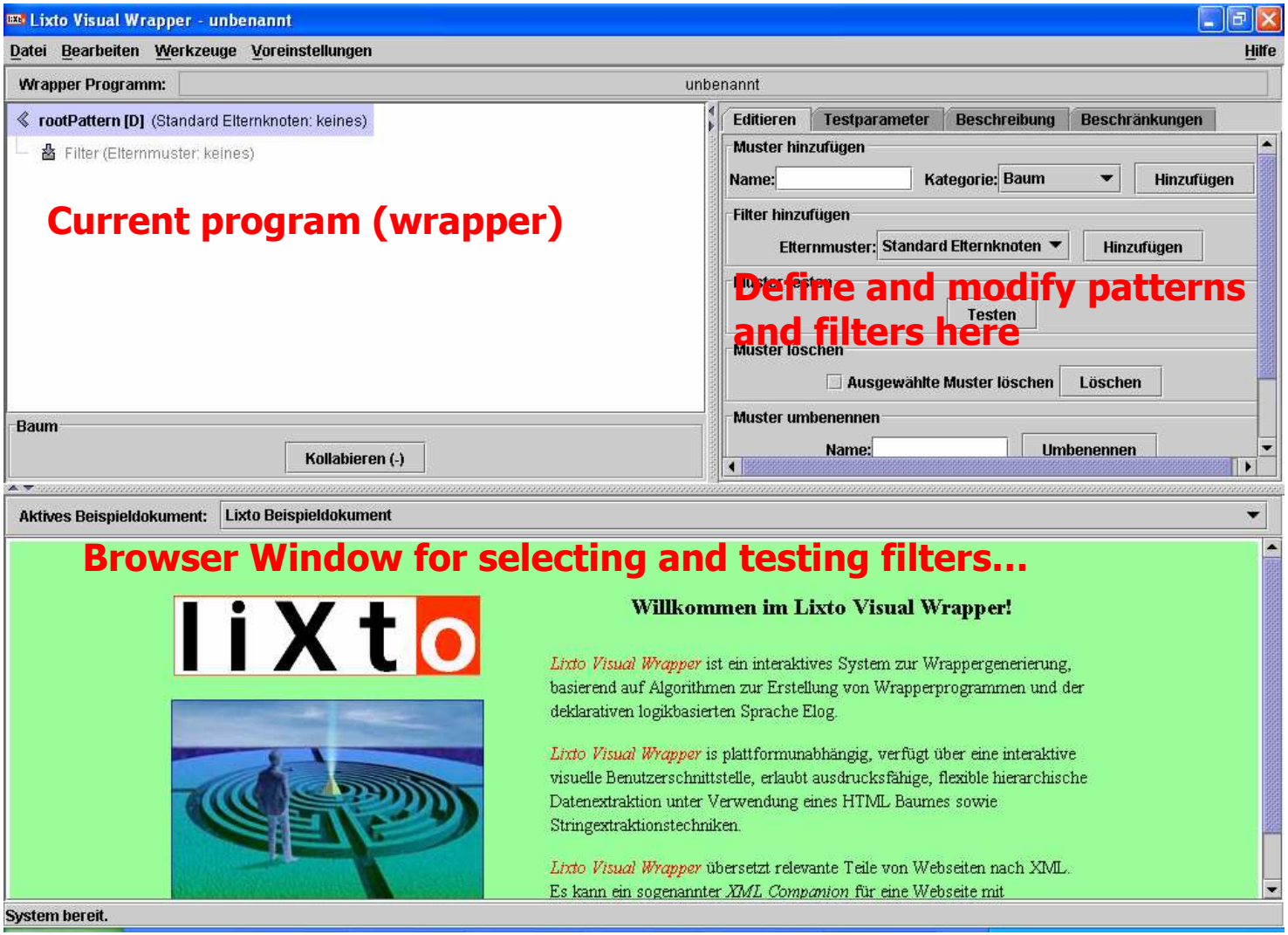
Different Patterns:

- Tree Patterns:
  - Use HTML properties and the tree structure for identifying relevant elements or element lists
- String Patterns:
  - operate on “flat” strings (e.g. divide first name and last name by regexps),
  - Mostly for HTML leaf elements
  - can also be used for “invisible” content (attributes)
- Document Patterns: Link to other documents:
  - Allow to navigate to further documents (e.g. via a “Next page” link in the page you are wrapping

LiXto: Add the URL of a page to wrap: The Document manager



Lixto: Patterns & Filters



Alternative to Wrapper Tools:

- Doing it from scratch, using APIs and JAVA… Ingredients:

- W3C Tidy:

  http://tidy.sourceforge.net/

  http://jtidy.sourceforge.net/

- XSLT processor e.g. Apache XALAN:

  Java: http://xml.apache.org/xalan-j/index.html

  C++: http://xml.apache.org/xalan-c/index.html

The solution particularly for Web integration:

- 2 alternatives:
  - Top-down: Create wrappers for current web sites and extract data automatically (wrappers)
    - *Today we mainly focus on this part!*
  - **Bottom-up: Instead of publishing natural language, publish machine-processable data directly (semantic Web idea!)!**

Towards a semantic Web:

- What's missing?
  Semantic Matching! An Example:
- An agent for the Internet Shopping domain:

  Program a KB-agent to search relevant pages for a certain query?

Q: What means relevant?

Generic Online Store

Select from our fine line of products:

- Computers
- Cameras
- Books
- Videos
- Music

```
<h1>Generic Online Store</h1>
<i>Select</i> from our fine line of products:
<ul>
<li> <a href="http://gen-store.com/compu">Computers</a>
<li> <a href="http://gen-store.com/camer">Cameras</a>
<li> <a href="http://gen-store.com/books">Books</a>
<li> <a href="http://gen-store.com/video">Videos</a>
<li> <a href="http://gen-store.com/music">Music</a>
</ul>
```

Example: What knowledge does an an agent for the Internet Shopping domain need?

"I'll find relevant offers at pages linked from OnlineStores which are linked via pages **relevant to my query** and which contain an offer"

"An offer is a page which contains an object relevant to my query and an **option to buy** or a **price.**"

Amazon 2 OnlineStores Æ Homepage(amazon,"http://www.amazon.com/")
Ebay 2 OnlineStores Æ Homepage(Ebay,"http://www.ebay.com/")
GenStore 2 OnlineStores Æ Homepage(GenStore,"http://www.gen-store.com/")

Relevant(page,url,query) , 9 store, home store 2 OnlineStores Æ Homepage(store,home)
Æ 9 url_2 RelevantChain(home,url_2,query) Æ Link(url_2,url) Æ page=GetPage(url)

RelevantChain(start,end,query) , start=end
Ç (9 u,text LinkText(start,u,text Æ RelevantCategoryName(query,text) Æ RelevantChain(u,end,query))

RelevantCategoryName(query,text) ,
9 c_1,c_2 Name(query,c_1), Name(text,c_2) Æ (c_1 µ c_2 Ç c_2 µ c_1)

The last logical expression says (informally) something like:
"An link is relevant to my query if it has a category which is a superclass or a subclass of my queried category in the link text."

Example: An agent for the Internet Shopping domain

- Taxonomy for product categories:

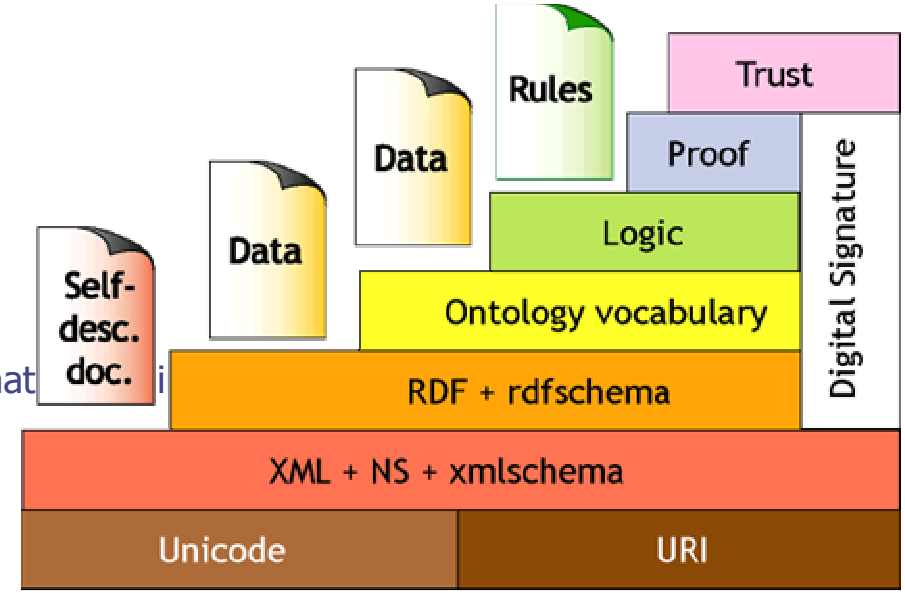
$Books \subset Products$	$Name(\text{“books”}, Books)$
$MusicRecordings \subset Products$	$Name(\text{“music”}, MusicRecordings)$
$\quad MusicCDs \subset MusicRecordings$	$\quad Name(\text{“CDs”}, MusicCDs)$
$\quad MusicTapes \subset MusicRecordings$	$\quad Name(\text{“tapes”}, MusicTapes)$
$Electronics \subset Products$	$Name(\text{“electronics”}, Electronics)$
$\quad DigitalCameras \subset Electronics$	$\quad Name(\text{“digital cameras”}, DigitalCameras)$
$\quad StereoEquipment \subset Electronics$	$\quad Name(\text{“stereos”}, StereoEquipment)$
$\quad Computers \subset Electronics$	$\quad Name(\text{“computers”}, Computers)$
$\quad\quad LaptopComputers \subset Computers$	$\quad\quad Name(\text{“laptops”}, LaptopComputers)$
$\quad\quad DesktopComputers \subset Computers$	$\quad\quad Name(\text{“desktops”}, DesktopComputers)$
$\quad\quad \ldots$	$\quad\quad \ldots$
(a)	(b)

- For successful search on the Internet, taxonomies of categories are important! Make the structure of knowledge available online! Ontologies!

Semantic Web

- Instead of publishing natural language, publish machine-processable meta-data directly (semantic Web idea!)!
- Provide standards on top of XML to describe the meaning of published knowledge
- This meta-data shall ideally also enable standardization of also the wrapping step.
- Provide the means to publish data on relations and taxonomies of data on the Web

- More on that i

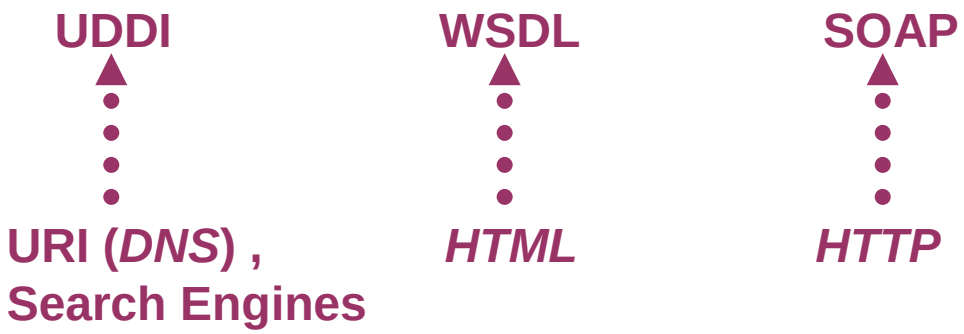


Web Services

- By annotating Web static Web pages with meta-data, we still haven't solved the issue about interaction with services on the Web.
- Web services standardize this interaction, i.e. standardize service integration like XML standardized data representation!

Web Services: SOAP, WSDL, UDDI

- SOAP: common protocol for message exchange
- WSDL: a language for defining insterfaces, partly based on XML Schema
- UDDI(Universal Description, Discovery and Integration): a repository and API standard for advertising and finding Web services.



- Later coming lectures…

References

- **XPath:** http://www.w3.org/TR/xpath
- **XSLT:** http://www.w3.org/Style/XSL/
- **LiXto:** http://www.lixto.com/ (only commercial, no downloads :-( )

- **More Wrapper Tools** (a bit outdated)
  http://www.wifo.uni-mannheim.de/~kuhlins/wrappertools/