Xyleme
Un Entrepôt de (toutes les) données XML
www.xyleme.com

Xyleme Group:
Verso/INRIA, Univ. Orsay,
Mannheim Univ., CNAM-Paris
The Web Today

• **Query** = keywords to retrieve URLs
  – Imprecise
  – Query results are useless for further processing
  – Important “handiwork” to go from the returned URLs to the required information

• **Applications:** based on ad-hoc wrapping
  – Expensive
  – Incomplete
  – Short-lived, not adapted to the Web constant changes
Corporate information environment

Web

- Ad-hoc applications written by web-experts tailored for specific tasks and data.
  - I.e. inflexible and expensive

Information System

- Manual searches using browsers
  - Manual updates
The Web Tomorrow

XML supersedes HTML

Xyleme revolutionizes the Web

- A repository for the world wide knowledge
- Semantic indexing and query processing
  - results that can be directly processed by applications
  - not just keyword search returning URLs, but queries returning information possibly spanning several pages
- Knowledge change control
- Environment to develop applications that integrate data from the web
Some Numbers

• Lawrence & Giles in *Nature* (July 99):
  – 800 millions pages (6 terabytes of data)
  – 3 millions public servers out of 16 millions

• Estimations (in *Nature*, May 2000)
  – 1 billion today
  – 100 billions in 2002 (??)
XML Estimation

• Assuming the same number of documents, with an average size of 10K, we have: $8 \times 10^{12}$ bytes (8000 Gigas)

• Recently, Alta Vista announced a new version indexing 400 M pages using 130 Alpha each with 16 GB of RAM.
  – I.e., data = indexes = $2 \times 10^{12}$ bytes
Percentage covered by Crawlers

Source: searchenginewatch.com
The Web Structure

Source: IBM, AltaVista, Compaq
Competition

• Search engines

• Sequoia XML indexer Xdex: small scale, very poor data integration, no subscription, no result construction, no store.

• XML repositories (e.g., Tamino from Software AG): small scale, no query language, no data integration, no subscription.
Xyleme: Some Numbers

• 4 millions of documents per day and crawler
• 1 million of XML documents can be indexed and loaded per day
Corporate information environment with Xyleme

Repository

Query Engine

Information System

Crawling & interpreting data

Web

publishing

Systematic updating

queries

searches
The Technical Team

• Researchers and PhD students from various labs and development groups:
  – INRIA (the Verso group)
  – U. Mannheim (the database group)
  – LRI, U. Orsay (the IASI group)
  – CNAM (Vertigo)
Xyleme Magic: XML
### The Integration Problem

<table>
<thead>
<tr>
<th>Ref</th>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>X23</td>
<td>Camera</td>
<td>359.99</td>
</tr>
<tr>
<td>R2D2</td>
<td>Robot</td>
<td>19350.00</td>
</tr>
<tr>
<td>Z25</td>
<td>PC</td>
<td>1299.99</td>
</tr>
</tbody>
</table>

The `<b>` X23 `</b>` new camera replaces the `<b>` X22 `</b>`. It comes equipped with a flash (worth by itself `<i>`53.99 $</i>), and provides great quality for only `<i>`359.99 $</i>.

---

Information System

---

**HTML**
XML Solution

<table>
<thead>
<tr>
<th>Ref</th>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>X23</td>
<td>Camera</td>
<td>359.99</td>
</tr>
<tr>
<td>R2D2</td>
<td>Robot</td>
<td>19350.00</td>
</tr>
<tr>
<td>Z25</td>
<td>PC</td>
<td>1299.99</td>
</tr>
</tbody>
</table>

Information System

XML

```xml
<product reference="X23">
  <designation>camera</designation>
  <price unit=Dollars>359.99</price>
  <description>…</description>
</product>
```

Ref ← product/reference
Name ← product/designation
Price ← product/price
<!ELEMENT product (designation, price, description) >
  <!ATTLIST product reference ID #REQUIRED>
<!ELEMENT designation PCDATA ) >
<!ELEMENT price PCDATA)>
  <!ATTLIST unit CDATA #REQUIRED>
...

Product.dtd
<xsd:schema xmlns:xsd=« http://www.w3.org/1999/XMLSchema » >

    <xsd:complexType name=« product »>
        <xsd:element name=« designation » type=« xsd:string »
            minOccurs=« 0 » />
        <xsd:element name=« price » type=« xsd:float »
            minOccurs=« 0 » />
    </xsd:complexType>

....

</xsd:complexType>
XML and Standard Browsers

![Diagram showing XML and Standard Browsers]
Challenges

- **Semantic**
  - Understand tags, partition the Web into semantic domains, provide a simple view of each domain
- **Dynamicty**
  - Control relevant changes in Web data
- **Scale**
  - Manage millions of XML documents and process millions of simultaneous queries
- **Efficiency!**
The Technology

- Functional Architecture
- Scaling: Machine Configuration
- Data Acquisition
- Change Control
- Semantic Data Integration
- Query Processing
- Repository
Xyleme: XML Dynamic DataWarehouse

- Store all XML resources of the Web
- Provide services on XML datas:
  - Query Processor
  - Semantic Classification
  - Data Monitoring
  - Native Storage
  - XML document Versionning
  - XML automatic or user driven acquisition
  - Graphical User Interface through the Web
Functional Architecture

- **Repository and Index Manager**
- **Change Control**
- **Query Processor**
- **Semantic Module**
- **User Interface**
- **Web Interface**
- **Xyleme Interface**
- **Acquisition & Crawler**
- **Loader**
- **Change Control**
- **Query Processor**
- **Semantic Module**
- **Repository and Index Manager**
Scaling

• Parallelism based on
  – Partitioning
    • XML documents
    • URL table
    • Indexes (semantic partitioning)
  – Memory replication
  – Autonomous machines (PCs)
    • Communication via Corba, caches are used for data flow
Architecture

I N T E R N E T

Interface Change | Semantic Global Query

Web Interface Crawler Global Loader

Loader | Query | Version Repository

DTDi,DTDj XML DOC extent

Index

Index

Index

Loader | Query | Version Repository

DTDk,DTDl XML DOC extent

DTDm, .. XML DOC extent

DTDp ... XML DOC extent
Data Acquisition
Data Acquisition

• Xyleme Robot visiting the Web: « Xyro »

• Visit HTML/XML pages
  • HTML pages may lead to XML pages

• Standard search to find new pages
  • bounded depth search

• Sophisticate search/refresh strategies based on
  • importance ranking of pages
  • change frequency and age of pages
  • publications (by owners) & subscriptions (by users)
Page Life Cycle

Internet

Web Interface (Crawler)

MetaData Manager
MD5, last access, ...

Page Scheduler

Page Importance
à la Google
Web Interface : Xyro

- follow links in HTML pages (on the fly parsing)
- send XML pages to loader
- hash table to detect known pages
- submissions from Xyleme with priorities
- 4 millions pages per day
- standard PC (450 MHz, 128 M memory)
Metadata Manager

- Stores metadata on HTML/XML pages
  - URL ↔ URL-ID [integer identifying the page]
  - temporal data
  - document signature
  - list of children
  - type of documents (HTML, XML, DTD)
  - more
Metadata Manager

- For each page that is crawled
  - update the metadata of the page
  - update the Link matrix
  - about 30 random accesses to obtain the URL-IDs of the children
- Main memory map URL ↔ URL-ID
- About 5 millions URL/hour
Refresh Policy

Based on:

– importance of the page: weight
– staleness of the page
– estimated change rate of the page
– publication or subscription requests
Page Importance

• Two main uses:
  – classify results (XML) by importance [Google]
  – guides the acquisition of pages (XML, HTML)

• *Intuition: a page is important if many important pages reference it:* FIXPOINT

• Alternative that we will not consider:
  – number of *useful* accesses to the page biased, difficult to evaluate
Page Importance: Algorithm

- Algo: Fixpoint
  - $W^0(k) = 1/N$ (initialization)
  - $W^m(k) = \Sigma_i [M(i,k) * W^{m-1}(i)/OUT(i)]$

- Problem: Size (+100 millions of pages)
  - $10^{16}$ entries in the Matrix
  - very sparse matrix

⇒ even that is too large to fit in main memory
Page Importance: Implementation

• The matrix is very sparse
  – $M(i,-)$ is stored as a list
  – computation of $W^m$ (line/line)

    for $i = 1$ to $N$ do
    
    [ read $M(i,-)$; process the line ]

• In main memory: $W^m$, $W^{m-1}(i)$ and $M(i,-)$

  Big!
Page Importance : Implementation

\[ W^m = M(i,-) \times W^{m-1}(i) \]
Page Importance/Usefulness

- XML pages: importance
- HTML pages: usefulness to discover new XML pages another fixpoint

Importance of pages

Importance and usefulness of pages

Importance and usefulness of pages with pub/sub
Page Importance: Conclusion

• Techniques for fixpoint convergence
• Some results
  – convergence is fast (≈OK after 10)
  – simple precision suffices
  – possible on a standard PC
  – parallelism possible
  – other optimization studied
Evaluation of Change Rate

- Based on the Last Date of Change [provided by HTTP header of the page]
  - in general reliable but ...

- Based on the number $M$ of changes detected the last $N$ times the pages was refreshed
  - limits: do not know the actual number of changes

*First one more precise*
Page Scheduler: General Approach

• Refresh cycle [6 hours in experiments]
• Circle the list of known pages
• Based on available data decides for each page whether it should be read or not
• Decision based on the minimization of a cost function
Cost function

Based on:

• Staleness of the page
  – penalty for being out of date
    (at least one update missed)
  – penalty for being very old (many missed)

• Importance of the page

• Customer requests

Constraint: the bandwidth of the crawler
  (G pages per refresh cycle)
Cost Function

• Determine $f_i$ [refresh frequencies] that
• Minimize
  $- \sum_{1 \ldots N} \text{cost}_i(f_i)$
• Under the constraint
  $- \sum_{1 \ldots N} f_i \leq G$
• where:
  $- \text{cost}_i(f_i)$ penalty for page $i$ based on the estimated change frequency, its importance and assuming a refresh frequency of $f_i$
Publication

• Allows to publish documents in Xyleme
• Either by email or by publication document on the Web
• http://www-rocq.inria.fr/verso/verso.pub
  – specifies what should or should not be in Xyleme
  – specifies some info about it: e.g., refresh automatic/explicit, frequency, …
  – others: access rights?
Tuning Parameters

- Specify refresh cycle time
- Assign resources to discovery vs refresh
- Assign resources to discovery XML/HTML (absolute parameter)
- Specify importance factor between XML/HTML pages (relative parameter)
Change Control
Change Management

• The Web changes all the time

• Data acquisition
  – automatic and via publication

• Monitoring
  – subscriptions
  – continuous queries
  – versions
Subscription

• Users can subscribe to certain events, e.g.,
  • changes in all pages of a certain DTD or of a certain semantic domain
  • insertion of a new product in a particular catalog or in all catalogs with a particular DTD

• They may request to be notified
  • at the time the event is detected by Xyleme
  • regularly, e.g., once a week
Continuous Queries

• Queries asked regularly or when some events are detected
  – send me each Monday the list of movies in Pariscope
  – send me each Monday the list of new movies in Pariscope
  – each time Xyleme detects that a new member is added to the Verso group, send me their lists of publications from their homepages
Versions and Deltas

• For some documents, store changes (deltas)
  – storage: last version + sequence of deltas
  – complete delta: reconstruct old versions
  – partial delta: allow to send changes to the user and allow refresh

• Deltas are XML documents
  – so changes can be queried as standard data
The Information Factory

Web

loaders

changes detection

documents and deltas

time

subscription processor

continuous queries

send notification

Repository

version queries

results
Support for Deltas

• **Diff algorithm**
  – allows to compute the difference between two consecutive versions of the same document

• **Naming scheme for XML elements**
  – one element is assigned a unique identifier for its entire life
  – compact way of representing/storing these IDs
Semantic Data Integration
Data Integration

• Several DTDs related to an application domain
  – heterogeneous vocabulary and structure
• Xyleme Semantic Module
  – gives the illusion that the system maintains an homogeneous database for this domain
  – abstracts a set of DTDs into a hierarchy of pertinent terms for a particular domain (business, culture, tourism, biology, …)

1 abstract DTD ⇔ 1 domain
Abstract DTD : Example

<table>
<thead>
<tr>
<th>Catalogue</th>
<th>Company</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name</td>
<td>Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
Abstract DTD = Interface

Many DTDs

Few Domains
Building Abstract DTDs

• Automatically
  – Cluster DTDs sharing similar « tags » using data mining techniques (frequent item sets) and linguistic tools (e.g., thesaurus, heuristics to extract words from composite words or abbreviations, etc.)
  – Use a thesaurus (e.g., Wordnet) to build a hierarchy from the « clustering tags »

• Manually
Building Abstract DTDs Automatically

- **Goal**: Discriminate several domains in a heterogenous set of DTDs
- **Problem**: DTD = set of words
  
  words are unreliable…

- **Solution**: group together the words dealing with the same concept → *Sense Units*
  
  ex : {author, writer, title, date} for the concept of *work*
Sense Units (example)

ISA

```
<us>
mathematics
solution
answer
result
atom
particle
mote
molecule
speck
course_of_instruction
course_of_study
course
class
resolution
<hyperonyme>
substance
human_activity
```

PART-OF

```
pure_mathematics
applied_mathematics
science
scientific_discipline
discipline
subject
field_of_study
study
branch_of_knowledge
knowledge_domain
knowledge_base
content
cognition
knowledge
psychological_feature
ion
isotope
monad
carbon_atom
hydrogen_atom
méthode
know-how
relation
</hyperonyme>
</hyperonyme>
chemical_element
element
molecule
elementary_particle
fundamental_particle
subatomic_particle
nucleus
meal
repost
</meronyme>
</antonyme>
</antonyme>
</couverture>
0.33
</couverture>
</us>

SYNONYM
DTD Clustering

• Provides the basis for building abstract DTDs
  – a cluster is characterized by a set of Sense Units

• Data mining techniques
  – Cluster DTD sharing similar « tags »
  – Handle synonymies, composed words, abbreviations

• Combines linguistic and statistical tools for detecting similarities: Wordnet, LSA
**Example**

<table>
<thead>
<tr>
<th></th>
<th>author</th>
<th>story</th>
<th>news</th>
<th>Wall-street</th>
<th>Dow-Jones</th>
<th>title</th>
<th>tale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc 1</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 2</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 3</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 4</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 5</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 6</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doc 7</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

**Clustering** is useless unless one knows the relationships between:

- story and tale
- Dow-Jones and Wall Street
Wordnet: Useful Relationships

• **Synonyms** ➢ One concept, two terms

• **Hypernyms / Hyponyms** ➢ two concepts linked through generalization/specialization
  - e.g., vehicle & car

• **Meronyms/Holonyms** ➢ two concepts linked through composition/inclusion
  - e.g., country & city
DTD Clustering : Method

- Filter words of DTDs
- Find Synonyms with WordNet
- Group these synsets with HAC, using distances given by document analysis (LSA)
  → Sense Units are obtained
- Reformulate the documents in terms of Sense Units
- Group together these documents with HAC, using cosinus distance
DTD Classification

• Input :
  - A set of abstract DTDs characterized by sets of Sense Units
  - A new DTD (or XML document) characterized by a set of words.

• Output :
  - The cluster which has enough Sense Units in common with the new document.
Search Algorithm

- Comparison between the set of words and the set of clusters:
  - Comparisons:
    - Combination of 3 comparisons $\Rightarrow$ a score.
  - « Bad » clusters elimination,
  - Reiteration until 0 or >1 candidate clusters,
    - For each iteration extension of the Sense Unit:
      - synsets$\rightarrow$hyponyms$\rightarrow$meronyms$\rightarrow$antonyms.
« Best » clusters election

- Scores Comparison:
  - Deletion of clusters which score is less than the score of the « best » cluster,
  - Preservation of clusters which have the most important score.
Abstract DTD => Interface

Catalogue

Company

Product

Name

Reference

Price

Description

Components
Principle of querying

User query

art/painting/work \Rightarrow d1/painting/name
\Rightarrow d2/work/title
\Rightarrow d3/painting
...

Union of Xyleme queries

MAPPINGS
Abstract DTD = Query Interface

Graphical User Interface

Abstract DTD

Mappings discovery
abstract DTD ↔ real DTDs

Reformulation

Query Processor

Querying

Real DTDs

Associated XML documents

Domain of interest
Generation of Mapping Rules

Set of real DTDs

How to connect the abstract knowledge with the concrete knowledge?

Abstract DTD

• Classified in a specific domain, ex: the culture domain
• Can be viewed as a set of paths

Mapping discovery process

• Description of the domain = tree of terms

Relationship between an abstract path and a set of concrete path
Automatic Generation : Term Similarity

If exist a similarity relationship between an element ’s name in a concrete DTD and a term in the abstract DTD.

Exemple :  
• Let the follow abstract path : Culture/Cinema/Movie  
• Let the follow concrete path : Film_Desc/Films/Film

Culture/Cinema/Movie ↔ Film_Desc/Films/Film

Similarities are :

• Syntactic, ex : Author & Author, Language & Lang, etc.
• Semantic, ex : Film & Movie (synonym), Artifact & Work Of Art (IS_A)
Automatic Generation: Interpretation Context

- Rule search is driven by the DTDs structure
  - notion of interpretation context: each node of a given path specify the interpretation context of its son

*Example*: Culture/Cinema/Movie/Title
Query Processing
Xyleme Query Language

A mix of OQL and XQL, will use the W3C standard when there will be one.

```
Select product/name, product/price
From doc in catalogue,
    product in doc/product
Where product//components contains “flash”
    and product/description contains “camera”
```
Principle of Querying

query on abstract dtd

- catalogue/product/price \(\Rightarrow\) d1//camera/price
- \(\Rightarrow\) d2/product/cost
- catalogue/product/description
- \(\Rightarrow\) d1//camera/description
- \(\Rightarrow\) d2/product/info, ref
- \(\Rightarrow\) d2/description

Abstract to Concrete Mapping  RULES

Union of concrete queries (possibly with Joins)
Query Processing

1. Partial translation, from abstract to concrete, to identify “machines” with relevant data

2. Algebraic rewriting, linear search strategy based on simple heuristics: in priority, use in memory indexes and minimize communication

3. Decomposition into local physical subplans and installation

4. Execution of plans

5. If needed, Relaxation
Execution Plans

A plan usually consists of:

1. parallel translation from abstract to concrete paths on the relevant index machines
2. parallel index scans to identify the relevant elements,
3. parallel construction of resulting elements.

Note: 2. Requires smart indexes
Structural and Full-Text Indexation

X ancestor of Y $\iff$
pre(X) < pre(Y) and
post(X) > post(Y)

X parent of Y $\iff$
X ancestor of Y and
level(X) = level(Y) - 1

Arbre Postfix/Prefix
Index Queries

Product

name
description

“camera”

(d1, 12, 200), (d1, 201, 400), ...

(d1, 11, 1), (d1, 224, 201), ...

(d1, 228, 237), ...

(d1, 228, 237), (d2, 14, 34), ...
Scaling

• Parallelism

• Use of semantic information to select only relevant machines

• Smart index structures to allow for efficient pipeline evaluation (i.e., no intermediate data structure)
Repository
Storage System

• NatixStore
  – efficient storage of trees in variable length records within fixed length pages

• Balancing of tree branches in case of overflow
  – minimize the number of I/O for direct access and scanning
  – good compromise: compaction / access time