Managing an XML warehouse in a P2P environment

Serge Abiteboul

INRIA and Xyleme

Outline

- Introduction
- Content warehouse
  - *A content warehouse: Xyleme*
- P2P-XML warehouse
- Issues in P2P-XML warehousing
  - *A language for distributed information exchange: Active XML*
- Very short conclusion

Introduction

Warehouse

- Goal: to provide an integrated access to heterogeneous, autonomous, distributed sources of information
- Main functionalities: acquire, transform, filter, clean and integrate data, support for queries
- Centralized access to information
- Warehouse vs. mediation
  - Warehouse: information is acquired in advance

  ≠ Mediation: information acquired when needed

Content vs. data warehouse

	Data warehouse	XML warehouse
Data	relational data numerical values	XML text
Enrichment	cleaning	cleaning, classification, semantics…
Integration and view	relations cube	XML
Query	SQL	Xquery, XSLT
Exploitation	OLAP statistical tools report generation	browsing report generation

Peer-to-peer

- A large and varying number of computers cooperate to solve some particular task without any centralized authority
- Goal: build an efficient, robust, scalable system based (typically) on inexpensive, unreliable computers distributed in a wide area network
- Examples
  - seti@home: search for extraterrestrial intelligence
  - kazaa: obtain free music/video over the net
  - cabal: decryption of 512 bits RSA code
  - grub: P2P Web search

An XML warehouse in P2P

- Warehouse: a very centralized system
- P2P: an ultra distributed system (no authority)
- P2P warehouse: *an oxymoron*?

No!

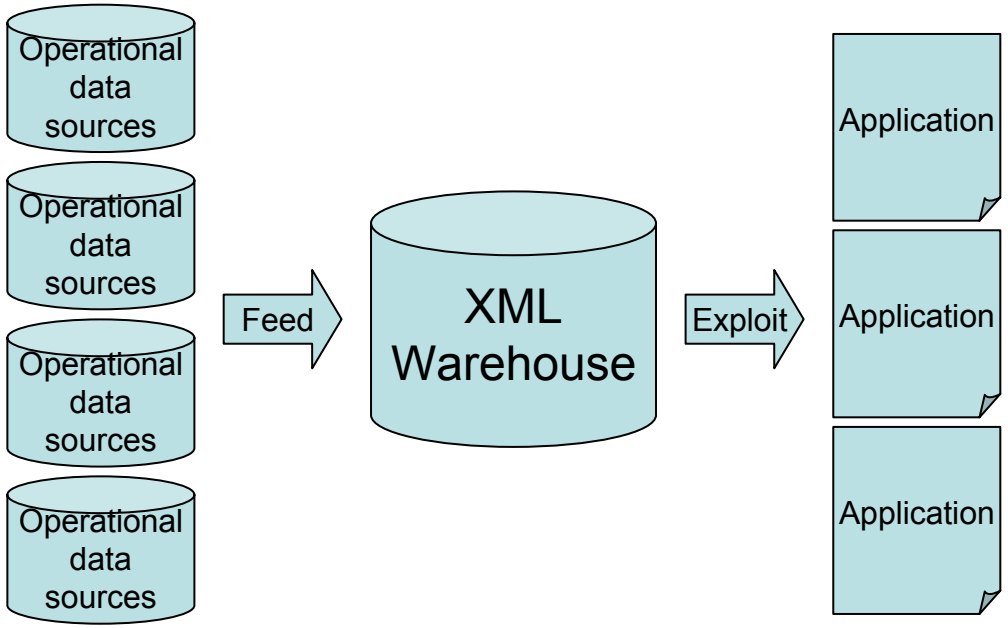
- A warehouse: from a logical viewpoint
- P2P system: from a physical viewpoint

Content warehouse

A general concept

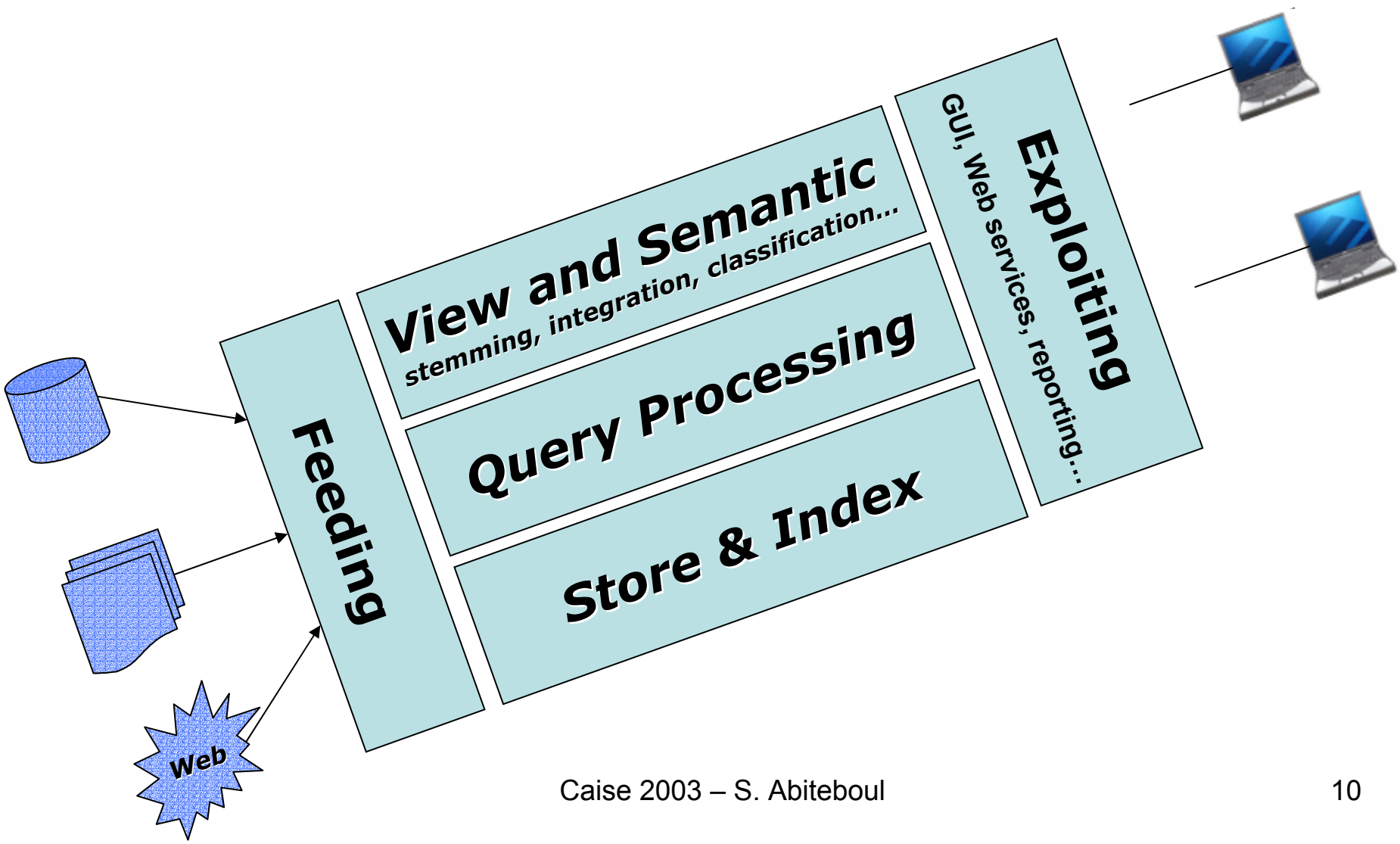
A precise example in mind: **Xyleme**

Warehouse



- Import data from many sources
- Add value to it without interfering with operational data
- Export integrated views of it

Functionalities



Functionalities: Feeding

- Loading from the Web (Internet and Intranet)
  - Web search
  - Web crawl
  - Access Web data via forms or Web services
- Plug-ins to load from
  - File systems, document management systems
  - Data bases, LDAP
  - Newsgroup, emails
  - Other applications
- Extraction and transformation
  - XSL-T or Xquery mappings for XML sources
  - XML-izers to load data from other formats
- Monitoring of the feeding

Functionalities: More feeding

- User feeding
  - Document editing
  - Meta data editing
- Publication

- *API: SOAP and WebDAV*

Functionalities: Storage

- Storage of (massive volume of) XML (terabytes)
- Indexing of (massive volume of) XML
  - By structure
  - By full-text
  - Linguistic support: multi language, stemming, synonyms, etc.
- Very efficient XML query processing
- Importance ranking
- Monitoring of the warehouse (support for subscriptions)
- Access control and security
- Versioning, archiving
- Recovery

- Possibly transaction mechanism

Functionalities: Enrichment

- Global organization
  - Global schema management
    - Management of collections
  - Incorporate domain ontologies and thesauri
  - Document classification
  - Cleaning by filtering out documents from collections, etc.
- Document enrichment
  - Concept extraction and tagging
  - Cleaning inside de document
  - Summarization, etc.
- Relationships between documents
  - Tables of contents
  - Tables of index
  - Cross referencing, etc.

Functionalities: View & integration

- View management
  - Document restructuring/mapping
  - Schema to schema mapping
- Semantic integration
  - Manual for complex ones and (semi-) automatic for simple ones
  - Tools to analyze a set of schemas
  - Tools to integrate them
  - Processing for queries on integration view
- Management of virtual data in a mediator style

Functionalities: Exploitation

- Access to the warehouse
  - Browsing
  - Querying by keywords, XPaths or Xquery
  - Temporal queries
- Query subscription
- Reporting
  - Generation of complex reports with pointers to documents, counts, abstracts…
  - Organized by collections, content, domains…

- *By GUI or from programs (Web service-based API)*

A Content Warehouse: Xyleme

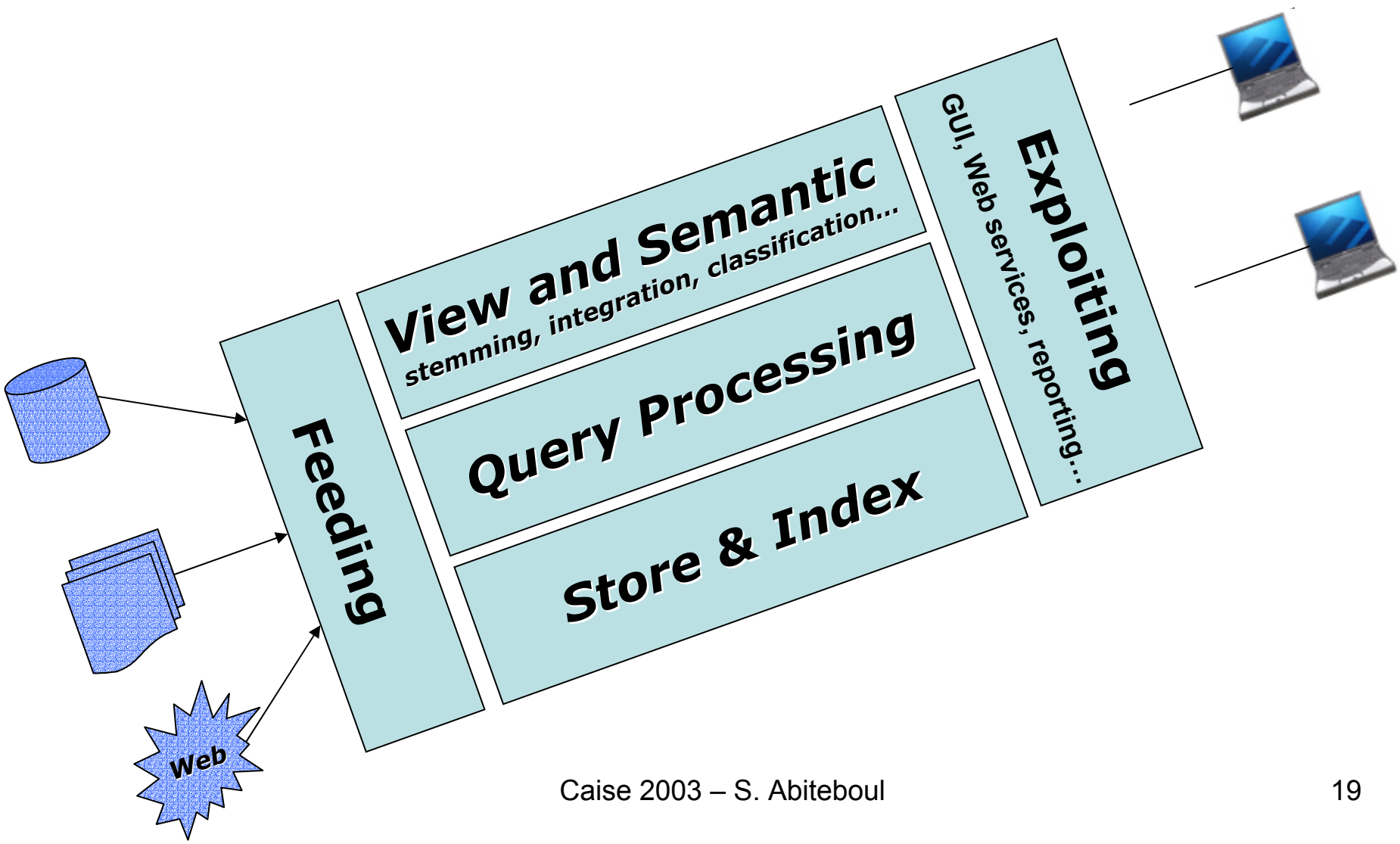
Xyleme – in short

- 1999: Xyleme research project at INRIA
- 2000: Creation of a spin-off
- 2003: About 30 people

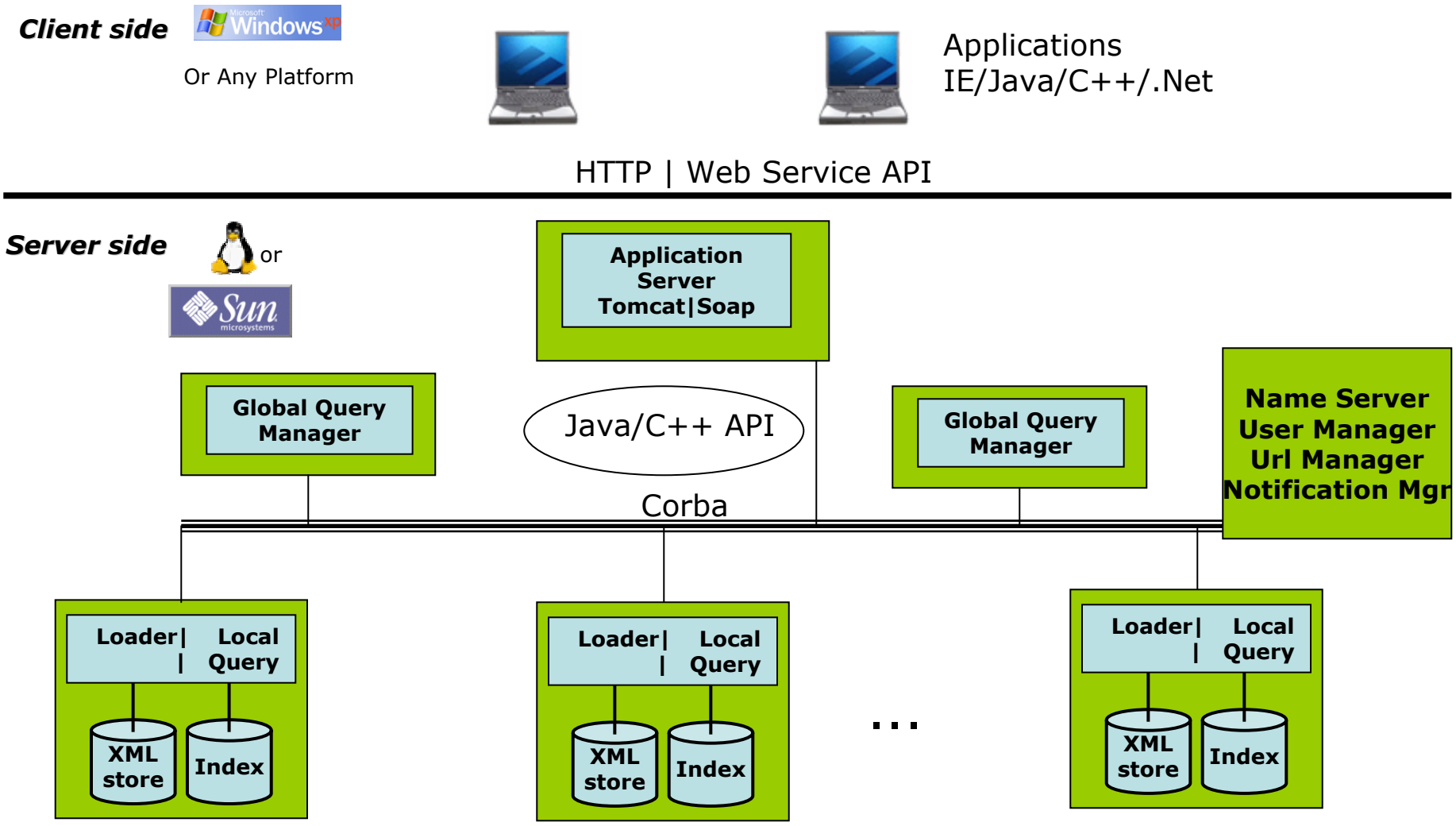
- Technology: a content warehouse built around a very efficient and scalable XML repository

- Application example: all articles of Le Monde in XML

Xyleme Functionalities



Xyleme Architecture

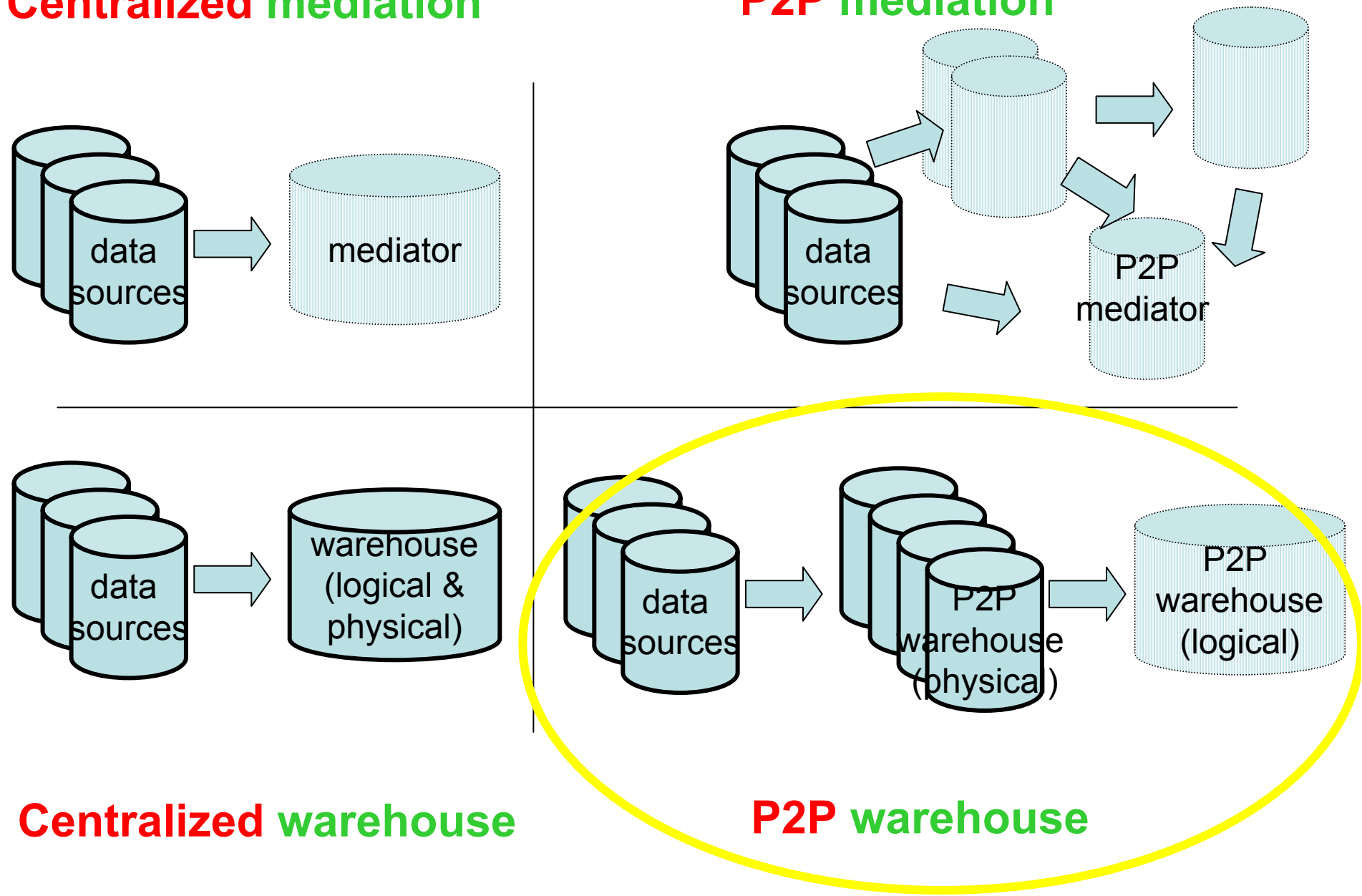


P2P-XML warehouse

2 dimensions

- Mediation vs. warehouse
  - Integration data is materialized or not
- Centralized vs. P2P
  - Integration system is centralized or not

- All cases: offer an entry point to access data from many sources


Centralized mediation
P2P mediation
data sources
mediator
data sources
P2P mediator
data sources
warehouse (logical & physical)
data sources
P2P warehouse (physical)
P2P warehouse (logical)
Centralized warehouse
P2P warehouse

P2P XML Warehouse

- Data sources and peers are distributed, transient and autonomous
- Information is distributed and replicated
- Nothing is centralized
  - Not the control, storage, indexing…
- The machines are “cooperating” with some level of trust to provide the functionalities of an XML warehouse

Example: preprints warehouse

- Each source provides scientific papers (preprints)
  - E.g., university labs
- Each WH peer stores scientific papers
  - E.g., dbINRIA and dbUCSD contain all preprints about database research
  - Other preprints of INRIA and UCSD are stored elsewhere
- Anybody can query any peer for any preprint
  - E.g., one can query dbINRIA for bioinformatics papers
- All sites are willing to use some common tools
  - Installation and linking of these tools should be 0-effort
- Advantages: reliability, timelessness, availability, performance, cost-effectiveness… (to be detailed)

Why distribute such a warehouse?

- Performance
  - Avoid bottleneck of centralized server
  - Replicate data locally and save on communications (caching)
- Ownership
  - Some peers may want to keep control over its own information (access control, access monitoring)
- Cost
  - Avoid the cost of a centralized server and take advantage of local resources (space and cycles)
  - Share cost of expensive operations
    - E.g., storage, query processing
    - E.g., web crawling

More advantages of distribution

- Reliability (via replication)
- Availability (via distribution and replication)
- Dynamicity
  - Allow peers to enter and leave the system in a transparent manner
  - Difficult to add/remove a new source of data in a centralized setting

Why not ?

- Performance
  - Complex queries over distributed collection may get expensive
  - Communication cost of queries
- Consistency maintenance
  - Keep copies in sync is complex and expensive
  - Difficult to support transaction
- Quality
  - Difficult to guarantee quality of service because of peer independence
- Availability
  - Difficult to guarantee because some peers may disappear resulting in unavailability of some information
  - Difficult to guarantee that no information will be lost

An opinion

- Very promising
- Very challenging
  - Can this work at the scale of the Web and millions of documents?
  - if we keep millions of documents in such a system, what is the probability that published today will still be available in 10 years, 100 years, 1000 years?

- Realistic first step
  - Some level of trust may be assumed from the peers
  - Enough peers are always available
  - Example: inside a big company

Related technology

- Data management on clusters
  - Google: indexing, web crawling, query processing
  - Xyleme: XML warehouse on a cluster of PC
- Distributed data management
  - Federated databases, etc.
- Network file systems
- P2P information processing
  - Look-up technology such as dynamic hash tables

Issues in P2P XML warehousing

P2P my favorite problem

P2P massive XML repository

- Xyleme is distributed over a cluster of PCs
- Here: wide area network
- New issues
  - Indexing
  - Distributed query processing

P2P Feed

- A particular feed (e.g., relational database) may be performed cooperatively between several peers
- Possible to split a feeding task
  - Load by one or more peers
  - Transform by one or more peers
  - Store in one or more peers
- Possible to replicate a feeding task

P2P Web engine

- Share the cost of Web crawling/indexing
  - E.g. engines in US, Europe…
  - Minimize the distance between engine and Web site
- Allow to crawl/index private portions of the Web
- One possible policy
  - Distribute the set of web sites between peers
  - Distribute the set of words to index between peers
  - Communications
    - Index information (word,page) to the site in charge of w
    - Page information (page) to the site in charge of page
    - More communications to maintain the graph of Web
    - Bufferize messages

P2P page ranking

- Google style
  - P2P maintenance of the graph of the Web
- Xyleme style [last W3 conf]
  - No need to store the graph
  - Communications between the crawlers to “move cash” around
- As usual in P2P systems: reliability issues
  - Trust: someone may cheat to increase the importance of some personal page
  - You trust the rating of Google, would you trust the ranking obtained by 100 000 peers you do not know
  - Replication, cryptographic techniques to verify the origin of “cash”

P2P Web mediation

- Centralized setting
  - Known correspondence/ontologies between information sources
- P2P setting
  - Need bridges between various sources
  - No global knowledge

- Some on-going works
  - Rousset+et, Halevy+al, Kementsiesidis+al

P2P Web Monitoring

- Centralized DBMS: triggers
- Web: monitoring
- Possible to factorize the effort by having a P2P monitoring system
  - Sources with triggering facilities
  - Other sources: share the work of regularly polling them
- Applications
  - Support for subscription queries
  - Web surveillance
  - Etc.
- Work on that [Sigmod01]

A language for distributed information exchange

- What is the exchange of information between the peers based on?
  - Low level protocols: XML and Web services
  - A high level language to query/exchange information
- We have a language for centralized and structured data: SQL
  - Solid foundations: relational calculus/algebra
- We need a language for distributed and semi-structured data
  - A proposal: Active XML
  - Warning: no serious foundation so far

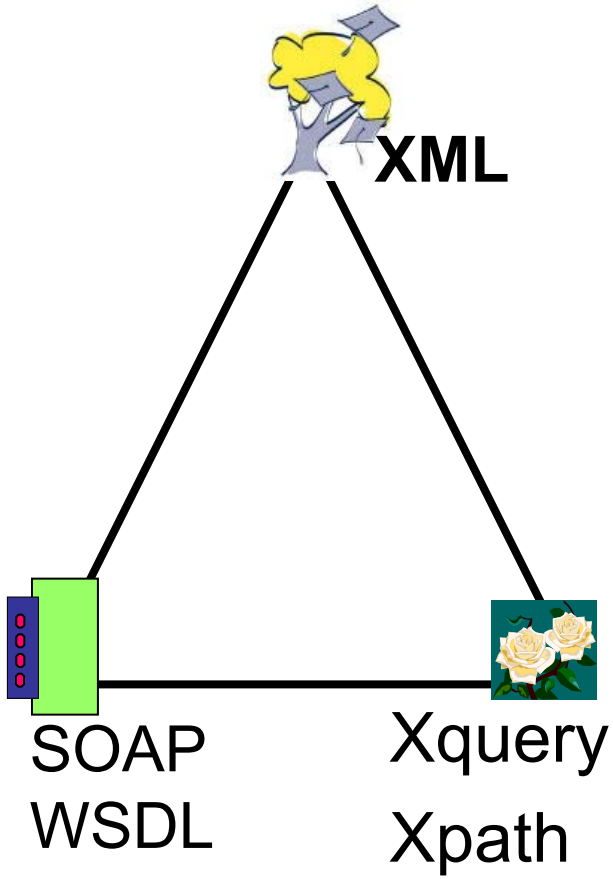
A language for distributed information exchange: Active XML

Joint work with:

Omar Benjelloun,
Bernd Amann,
Jerome Baumgarten
Angela Bonifati,
Gregory Cobéna,
Ioana Manolescu,
Tova Milo
and more

Preamble: The new context of distributed data management

- Standard for data exchange, XML
  - Extensible Markup Language
  - Labeled ordered trees
  - XML query languages: XPATH, Xquery
- Standards for distributed computing: Web services
  - SOAP, WSDL
  - Simple Object Access Protocol
  - Activation of methods on remote web servers



Active XML documents

- ***XML documents with embedded Web service calls*** (SOAP)
- Intensional
  - Some of the data is given explicitly whereas for some, its definition (i.e. the means to acquire it when needed) is given
- Dynamic
  - If the external sources change, the same document will provide different information
  - Reaction to world changes

XML + embedded service calls (omitting syntactic details)

```
<resorts state='Colorado'>
  <resort>
    <name> Aspen </name>
    <scond> Unisys.com/snow("Aspen") </scond>
    <hotels ID=AspHotels > ….
    Yahoo.com/GetHotels(<city name="Aspen"/>)
    </hotels>
  </resort>   …
</resorts>
```

May contain calls
to any SOAP web service :
- e-bay.net, google.com…

to any AXML web services
- ***to be defined***

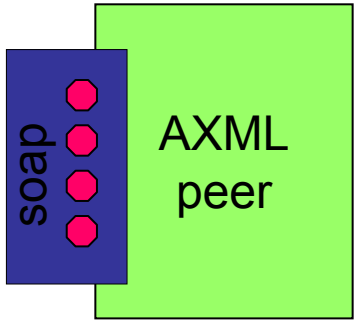
Example: AXML document after service evaluation

```
<resorts state='Colorado'>
  <resort>
    <name> Aspen </name>
    <scond> Unisys.com/snow("Aspen")
    <depth unit="meter">1</depth>
    </scond>
    <hotels ID=AspHotels > ….
    Yahoo.com/GetHotels (<city name="Aspen"/>)
    </hotels>
  </resort>…
 </resorts>
```

Not a new idea in databases
Not a new idea on the Web

- Mixing calls to data is an old idea
  - Procedural attributes in relational systems
  - Basis of Object Databases
- In HTML world
  - Sun's JSP, PHP+MySQL
- Call to Web services inside documents
  - Macromedia MX, Apache Jelly

Active XML peer



- Peer-to-peer architecture
- Each Active XML peer
  - **Repository**: manages Active XML data with embedded web service calls
  - **Web client**: uses Web services
  - **Web server**: provides (parameterized) queries/updates over the repository as web services

The main novel issue: the evaluation of calls

- When to activate the call
- Where to find its arguments
- What to do with its result
- How long with the returned data remain valid
- What exactly to exchange: to-call-or-not-to-call

When to activate the call

- Explicit pull mode
  - Frequency: Daily, weekly, etc.
  - After some event: e.g., when another service call completed
  - This aspect of the problem is related to ***active databases***
- Implicit pull mode : Lazy
  - When the data is requested
  - Difficulty : detect that the result of a particular request may be affected by a particular call
  - This is related to ***deductive databases***
- Push mode
  - E.g., based on a query subscription; the web server pushes information to the client
  - E.g., synchronization with an external source
  - This is related to ***stream and subscription queries***

What exactly to exchange
(Sigmod03-exchange)

- A parameter of a call contains some service calls
- The result of a call contains some service calls
- Do we have to evaluate these calls before transmitting the data or not

Hi John, what is the phone number of the CEO of INRIA?

- *(33 1) 39 66 00 01*
- *Look in INRIA directory at Larrouturou*
- *Find his name at www.inria.fr then look on the directory*

When exchanging data: *to-call-or-not-to-call*

- Someone asks for information about Aspen
- Definition of an extension of XML schema that distinguish between Hotel* and () → Hotel*
- What is the expected type
  - … SCond:sct… Hotels: Hotel*

  Evaluate all calls and return result
  - … SCond:() → sct… Hotels: Hotel*

  Get the list of hotels that are not full and return result
  - … SCond:() → sct … Hotels: () → Hotel*

  Do not evaluate any call and return result

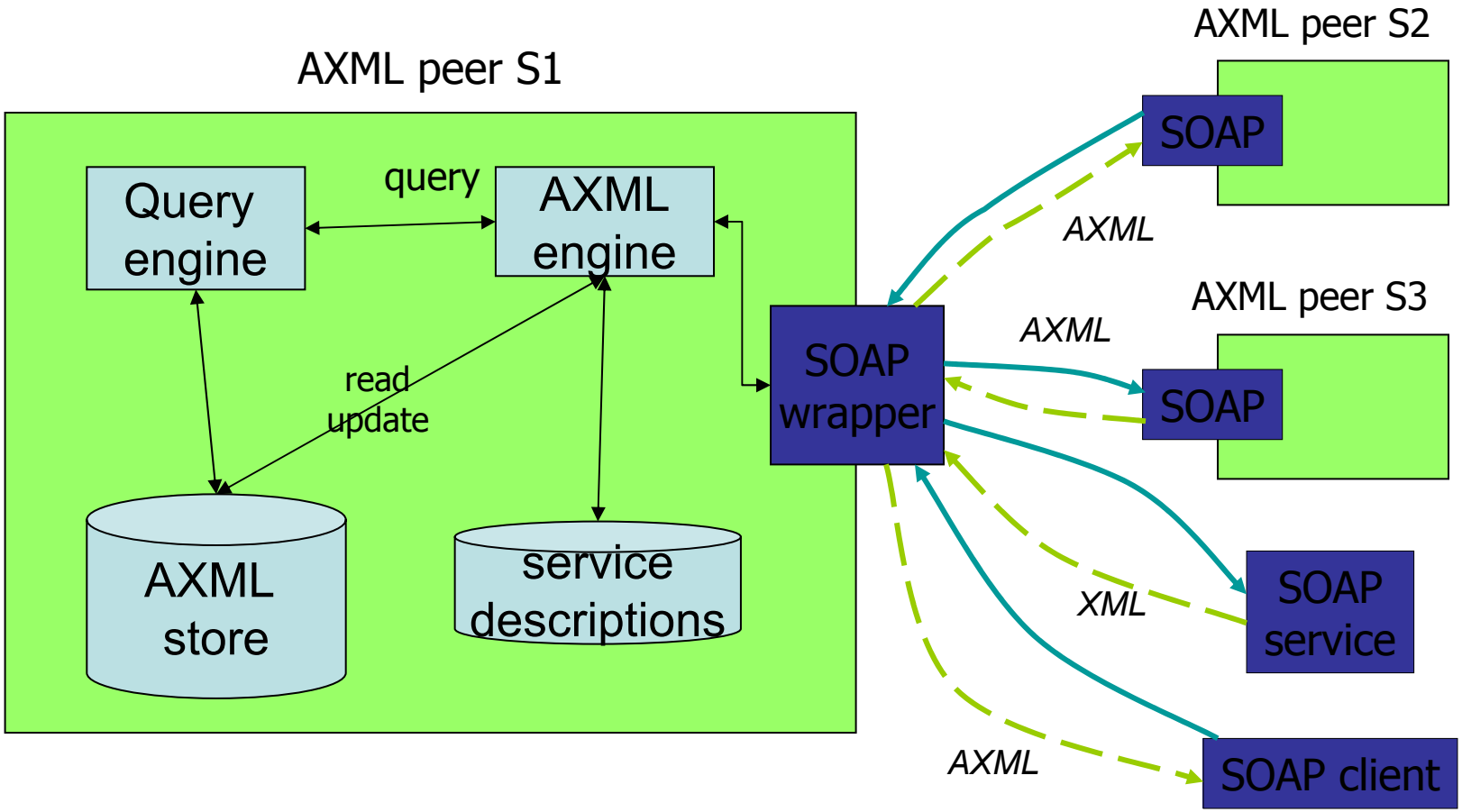
How is this controlled: typing

- This is based on a compromise between client and server
  - Server publishes a type for the service provided
  - Client publishes a type for the service expected
  - When sending a call, the client has to meet the requirements of the server
  - When receiving a call, the server tries to meet the requirements of the client
- General problem is undecidable [MSS]
- Algorithm under some restrictions

AXM peer as a server

- Publish query services over the repository in Xquery, XOQL, XPATH…
- Publish update services
- Provide/use continuous services (push)
  - Asynchronous services
  - Query subscription
  - Change control

Global architecture



Implementation

- SUN's Java SDK 1.4
  - XML parser
  - XPath processor, XSLT engine
- Apache Tomcat 4.0 servlet engine
- Apache Axis SOAP toolkit 1.0
- X-OQL query processor
  - persistent DOM repository
- JSP-based user interface
  - JSTL 1.0 standard tag library
- V0 – demo at VLDB'02
  - P2P auctioning system

Examples of applications

- Peer-2-peer auction [VLDB2002]
- Mobile computing [EC project Dbglobe]
- Web warehousing [French project e.dot]
- Network configuration
- Ambient computing [proposal air@large]

On going work

- On distribution and replication (Sigmod03-distrib)
- On security
- AXML on a telephone/pda

Very short conclusion

P2P content warehouse is not an oxymoron

- Many advantages
- Leads to revisiting all functionalities of content warehouses
- Let's do it
- Try Active XML

merci