The XCDE Library

Xml Compressed Document Engine

Paolo Ferragina

Andrea Mastroianni

Dipartimento di Informatica, Univ. Pisa

Great opportunity for IR…

Queries may exploit the tag structure to refine, rank and specialize the retrieval of the answers. For example:

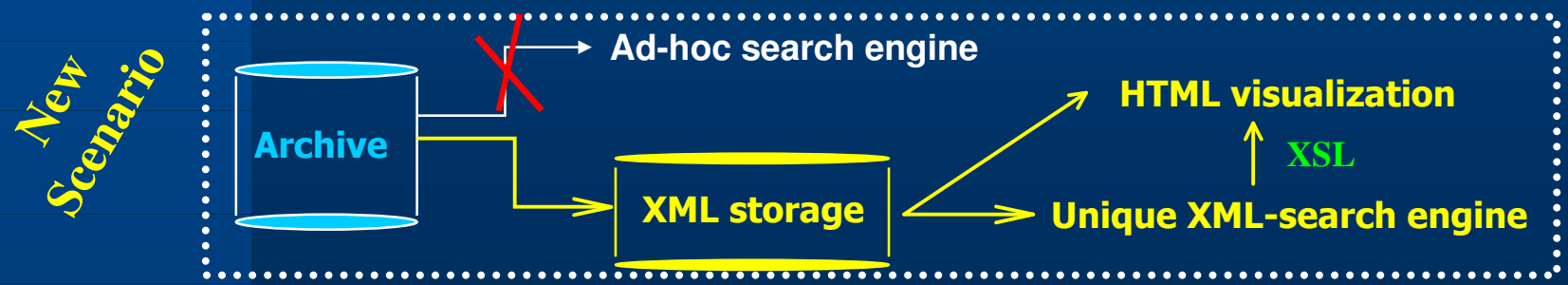
- **Proximity refined by exploiting the text structure**

  <author> ivo rossi </author> <author> ugo verdi </author>

- **Word disambiguation driven by tag names**

  <author> Brown … </author> *<university> Brown … </university>*
  <color> Brown …. </color> *<horse> Brown … </horse>*

- **Context extraction driven by "tag enclosure" → well-formedness**



XML storage: four approaches

- **Flat** : just file storage and processing routines (DOM or SAX)
  - Very slow and memory demanding in the DOM case
- **IR oriented** : full-text search capabilities
  - Forget the existence of text structure
  - Tag queries solved by searching for <tag
  - Proximity exploited to solve queries on text + structure
- **Database oriented** : built on top of existing DBMS
  - Relational or object-oriented
- **XML native** : aim at providing XML storage support

XML native storage

The literature offers various proposals:

- **TReSy**: String B-tree → large space occupancy (1997)
- **Xset**, **Bus**: build a DOM tree in main memory at query time
- **XYZ-find**: B-tree for storing pairs <path,word>
- **Natix**: DOM tree is partitioned into disk pages (see e.g. Xyleme)
- Some commercial products: **Tamino,…** (no details !)

✓ **Fabric**: Patricia tree for indexing *all* possible paths

✓ **XISS**: Inverted indexes + numbering scheme for doc structure access

✓ A plethora of other proposals….

"history"

Three interesting issues…

1. Space occupancy is usually not evaluated (it is high for DB-approaches)
2. Data structures and algorithms forget *known* results
3. No software in the form of a library for public use and development

Our project

XCDE Library: Requirements

- **XML documents may be:**
  - strongly textual (e.g. linguistic texts)
  - may occur without a DTD
  - retrievable in their original form (for XSL, browsers, post-processing…)

- **The library should offer:**
  1. Minimal space occupancy (doc + index ~ original doc size)
     ⇒ space critical applications: e.g. **e-books, Tablets, PDAs**,…
  2. State-of-the-art algorithms and data structures
  3. XML native storage for full control of the performance
  4. Flexibility for extensions and software development

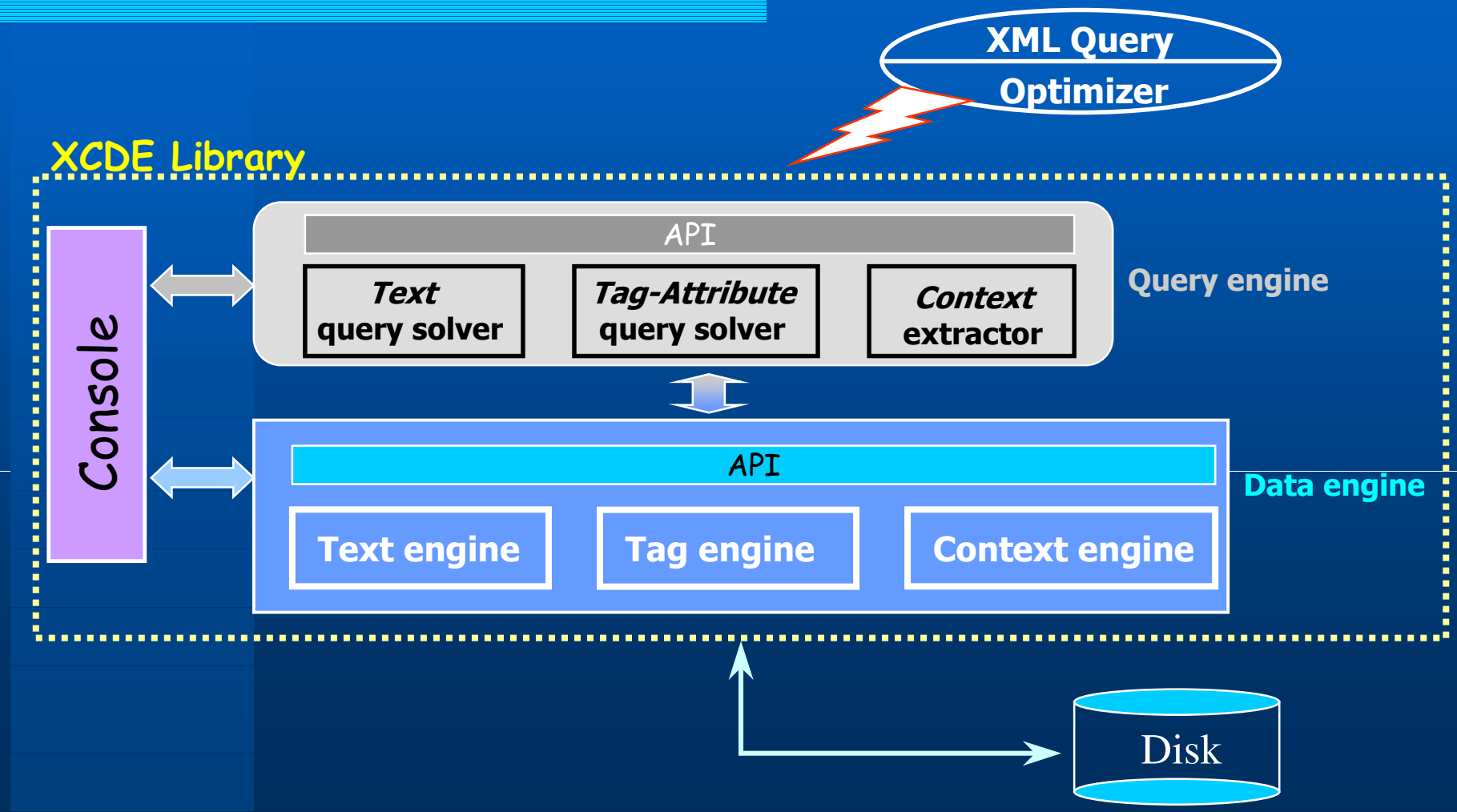
XCDE Library: Design choices

- **Single document indexing:**
  - Simple software architecture
  - Index customizable on each file (they are heterogeneous)
  - Ease of management, update and distribution
  - ❖ Blocking *via XML tagging* to speed up query

- **Full-control over the document content:**
  - Approximate or Regexp match on text or attribute names and values
  - Partial path queries, e.g. //root_tag//tag1//tag2, with distance

- **Well-formed context extraction:**
  - for rendering via XSL, Braille, Voice, OEB e-books, …

XCDE Library: The structure



Basic features

- **Main features:**
  - **Developed in C, tested under RedHat Linux 7.2 e Apache 1.3**
  - **Uses two public libraries: Zlib e Expat**
- **User may customize**
  - **definition of words and separators, including entities**
  - **set of data structures to be built**

✓ **Available for research and teaching purposes**

Not limited to Linux …

provided that those libraries are supported by the OS !!

Data engine: The document storage

- **We keep a Two-Level View:**
  - **Phisical view → original ASCII file**
  - **Logical view → file with internal entities expanded**

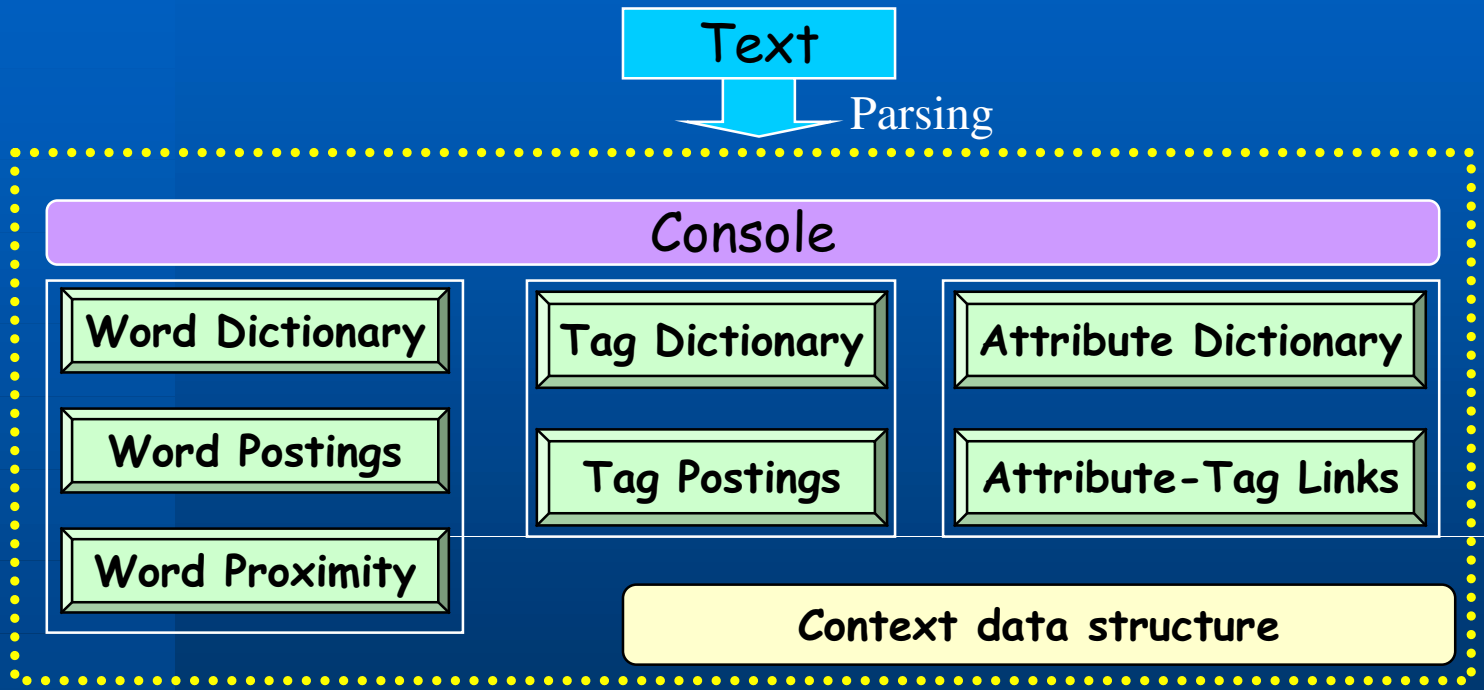
```
<!ENTITY unipi   "University of Pisa">
...
Paolo Ferragina, &unipi;   →  Paolo Ferragina, University of Pisa
```

- **Original text compressed twice by**
  - **Byte-aligned tagged Huffword → keep word identities**
  - **Gzip on blocks of 16Kb size → exploit phrase repetitions**

This ensures that …

the compressed file occupies about 33% of its original size !

Data engine: The indexes



- ✓ Dictionary compression by Gzip
- ✓ Postings compression by Continuation-bit

} less than 65 % of original file size !

Data engine: Context data structure

- **A special data structure:**
  - **Succinct space + fast access to doc structure**
    - **Space = Θ(#tags)**
    - **Context queries in O(1) time**

less than 1 % of the original file size

```
<?xml version="1.0" ?>
...
<weather-report>
    <date> 25/12/2001 </date>
    <time> 09:00 </time>
    <area> Pisa, Italy </area>
    <measurements>
        <skies> sunny </skies>
        <temp scale="C"> 2 </temp>
    </measurements>
</weather-report>
...
```

queried position

well-formed context extraction

Query engine: Some details

- **Text query supports :**
  - Standard word queries: prefix-match, suffix-match, substring-match;
  - Complex word queries: Approximate match, Regular expressions;
  - Proximity search.

- **Tag-Attribute query supports :**
  - Arbitrary path expressions, e.g. //doc/chap/*/page/*/line
  - All tags that enclose a given text position;
  - Fast computation of distances in tag nesting;
  - Complex searches on attribute values, and/or tag-attribute pairs.

As a result …

✓ We can implement Xpath, and "most of" Xquery !

✓ Space ~ original file size, very fast arbitrary IR or path queries

(from 70% to 80% due to dictionary access !!)

XCDE - XML Compressed Document Engine - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://sbrinz.di.unipi.it/~xcde/xcdelib.html

XCDE

Xml Compressed Document Engine

What is it ?

The XCDE Library is a native system to compress and index XML files. It was written in C at the Computer Science Department of the University of Pisa by Paolo Ferragina and Andrea Mastroianni. The library includes an API that allows users to store, index and compress XML documents and some commands (written using the API) for implementing higher-level queries and/or for document (de)compression operations. *State-of-the-art algorithms* and *data structures* were adapted to reach good *time/space* performance and to support efficiently some innovative features, as the extraction of well-formed portions of XML documents (*snippets*), proximity queries on words and *structural queries*. The library is modular, easy to use, and efficient. It's not public domain, but free for non-commercial purposes and *it comes with source*.

Overview of the library

The main characteristics of the library are the following:

- native: it operates at the bottom level just upon the File System. This solution allowed designers to control all the details of the implementation, using specific algorithms and data structures by which it is possible to obtain better *time/space* performance;
- modular: it was designed to ease future changes and expansions. The interface of the library is composed by C functions that implements access operations;

Done Internet

An example of use

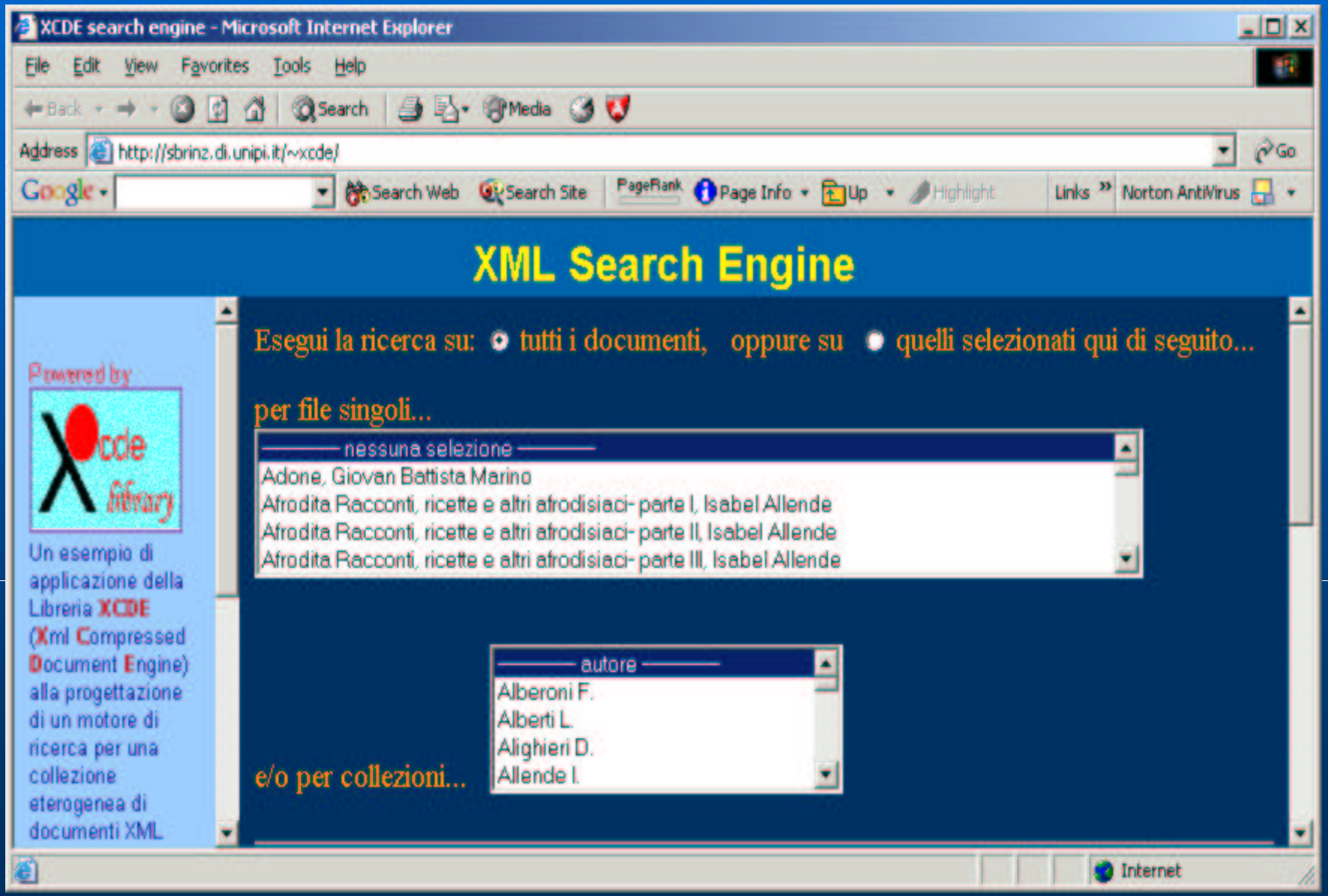
- **Collaboration with CIBIT** **(Biblioteca Italiana Telematica, Prof. Tavoni)**
  - **About 1500 texts marked with TEI-XML**
  - **Texts have strong linguistic content and are highly structured**
  - **Attribute values are complex strings with letters and numbers**

- **Design requirements:**
  - **IR-queries on words and attribute values**
  - **Queries may involve groups of files, specified on-line**
  - **Well-formed context extraction for customized visualization**

What we did ?

We used the XCDE lib to implement a simple XML search engine !

http://sbrinz.di.unipi.it/~xcde



http://sbrinz.di.unipi.it/~xcde

XCDE search engine - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://sbrinz.di.unipi.it/~xcde/

XML Search Engine

Powered by Xcde library

Un esempio di applicazione della Libreria XCDE (Xml Compressed Document Engine) alla progettazione di un motore di ricerca per una collezione eterogenea di documenti XML prodotti e messi a disposizione dal CIBIT.

Autori:
Paolo Ferragina
Andrea Mastroianni

	Tag	Attributo		Valore	
#1.	foreign	lang	Esatto	la	Esatto
#2.			Esatto		Esatto
#3.			Esatto		Esatto

Proximity (# parole)

Ambito della ricerca: Corpo del documento

Pattern 1: re — tipo occorrenza: Prefisso

Pattern 2: "militar[ie]" — tipo occorrenza: Espressione regolare

Pattern 3 — tipo occorrenza:

—specificare—
Parola
Sottostringa
Prefisso
Suffisso
1 errore
2 errori
Espressione regolare

Submit Query

Done — Internet

http://sbrinz.di.unipi.it/~xcde

XCDE Search Engine - Risultati: Libri della famiglia - Leon Battista Alberti - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Media

Address http://sbrinz.di.unipi.it/xcde-cgi-bin/extract.pl Go

Google Search Web Search Site PageRank Page Info Up Highlight Links Norton AntiVirus

occorrenza 1 (frame sotto)(nuova finestra)

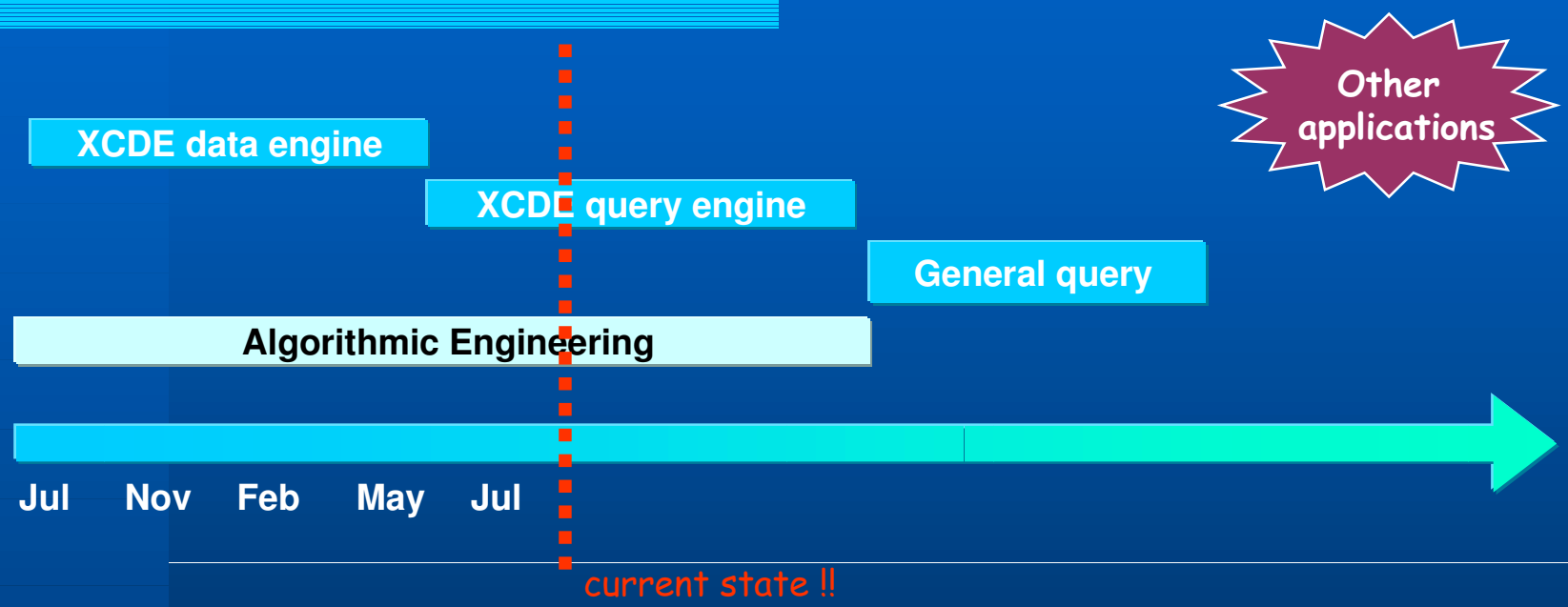
...

disputava de re militari, rispose

aperta coniunzione ed equalità d'animo, alcune con minor vinculo collegate e solo con domestichezza, conversazione e convivere, uso d'amicizia, contenute; quali tre e' nomina la prima naturale, l'altra equale, l'ultima ditta da quella antica consuetudine ch'e' cittadini di qui divertivano a casa quelli là, e' quali si riducono simili qui ospiti apresso di costoro, e per questo s'appella ospitale. Queste adunque simili scolastice e definizioni e descrizioni in ozio e in ombra fra' litterati non nego sono pur ioconde, e quasi preludio come all'uso dell'arme lo schermire: ma a travagliarsi in publico fra l'uso e costume degli uomini, se null'altro aducessero che sapere se la madre più che 'l padre ama e' nati suoi, o se l'amor del padre verso e' figliuoli sia maggior che quello de' figliuoli verso el padre, e qual cagion faccia e' fratelli insieme amarsi, temo loro interverrebbe come a quel Formio peripatetico filosofo, al quale Annibal, udita la sua lunghissima orazione dove e' disputava de **re militari**, rispose avere veduti assai, ma non alcuno pazzo maggior che costui, el quale dicendo forse stimasse potere in campo e contro all'inimici quanto in scuola ozioso disputando. E ben sai, in tanta diversità d'ingegni, in tanta dissimilitudine d'oppinioni, in tanta incertitudine di volontà, in tanta perversità di costumi, in tanta ambiguità, varietà, oscurità di sentenze, in tanta copia di fraudolenti, fallaci, perfidi, temerarii, audaci e rapaci uomini, in tanta instabilità di tutte le cose, chi mai si credesse colla sola simplicità e bontà potersi agiugnere amicizia, o pur conoscenze alcune non dannose e alfine tediose? Conviensi contro alla fraude, fallacie e perfidia essere preveduto, desto, cauto; contro alla temerità, audacia e rapina de' viziosi, opporvi

Done Internet

Plan of future activities



Future research …

- Exploit state-of-the-art compressed indices for *dictionaries;*
- Extend XCDE to manage highly populated collections;
- Implement a more powerful query language and a user-friendly interface;
- Study XML ranking, clustering, doc-struct encodings, fast path or tree queries.

A glimpse onto XML features

- **Text-based tag language, ~ HTML but data oriented.**
- **Ground rules on XML syntax:**
  - **Ease document parsing and processing;**
  - **Self-describing mark-up → represent any kind of data.**
- **It is platform independent.**

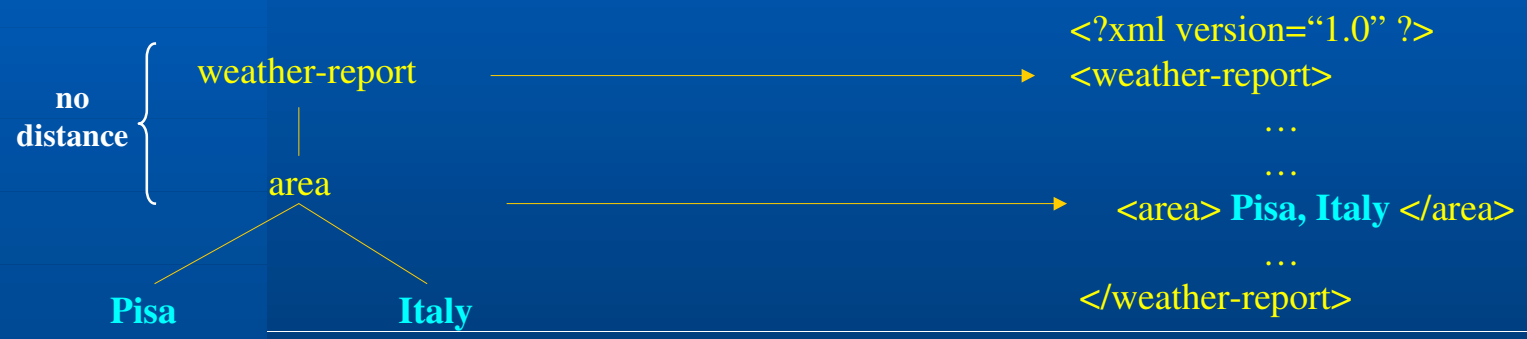
" What you see is ~~what you get~~ " (~~WYSIWYG languages~~)
all you've got !! (B. Kerningan)

Allows "write once, distil anything, publish everywhere" (XSL)

An example of query

- **How do we find the *area* of *Pisa* in the *weather-report* document ?**

--tag --xml_exact weather-report --tag --xml_exact area --word --xml_exact **Pisa** --word --xml_exact **Italy**



- **Or more precisely:**

--tag --xml_exact weather-report --xml_dist 1 --tag --xml_exact area
--proximity 1 --word --xml_exact **Pisa** --word --xml_exact **Italy**

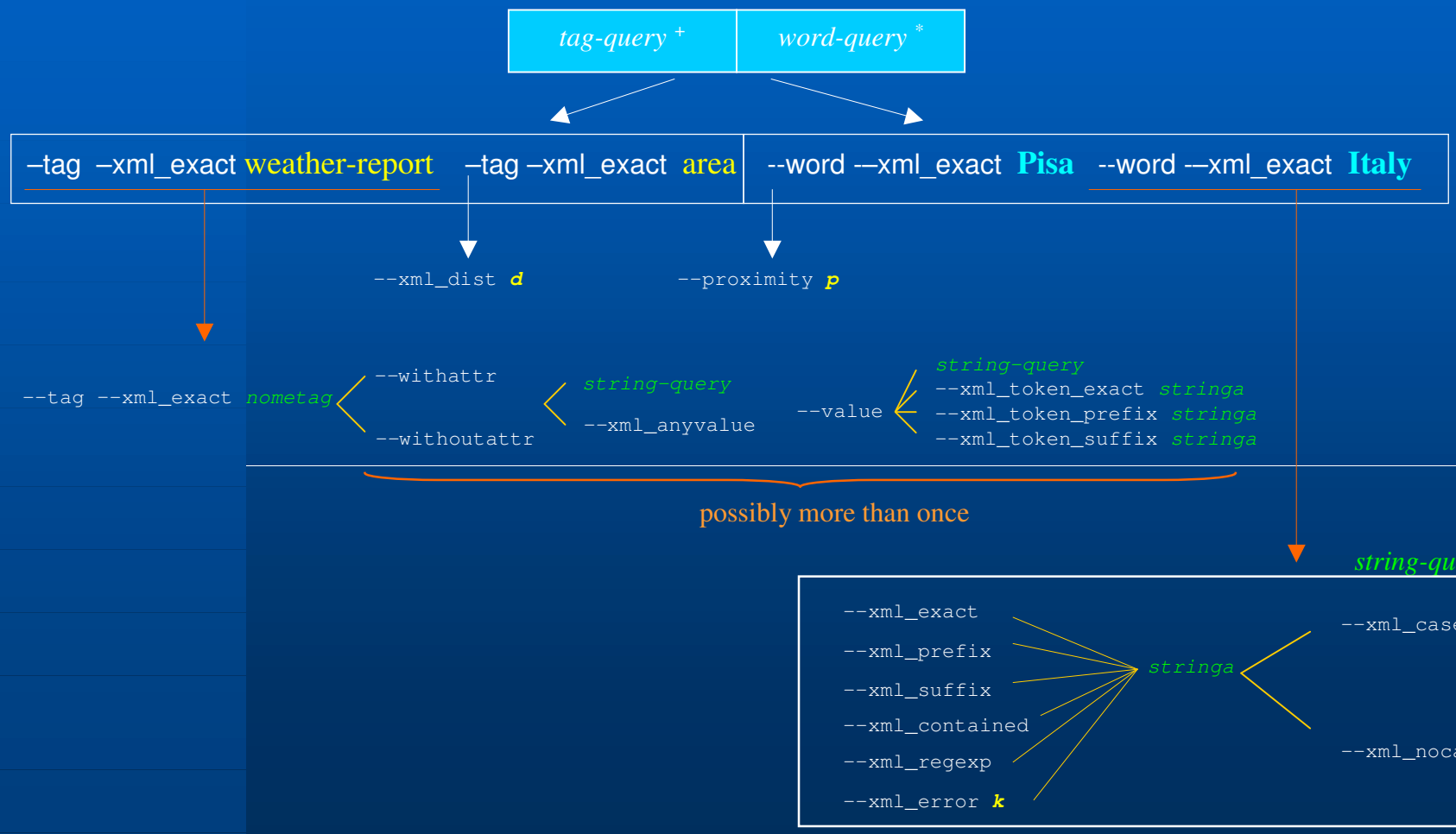
DBMS and XML

- **Main idea:**
  - Represent the document tree via tuples or set of objects
  - Query engine use standard *join* and *scan*
  - Some additional indexes for special accesses (e.g. Fabric)

- **Some advantages:**
  - Standard DB engines can be used without migration
  - Query languages are well known: e.g. SQL
  - Query optimisers are well tuned

- **Some disadvantages: twofold mapping for storage and query !**
  - Query navigation is costly, simulated via many joins
  - Query optimiser looses knowledge on XML nature of the document
  - Number of tables is high and much space is wasted

The query language in a "nutshell"



DBMS and XML

- **Main idea:**
  - Represent the document tree via tuples or set of objects;
  - *Select-from-where* clause to navigate into the tree;
  - Query engine use standard *join* and *scan;*
  - Some additional indexes for special accesses;

- **Advantages:**
  - Standard DB engines can be used without migration;
  - OO easily holds a tree structure;
  - Query language is well known: SQL or OQL;
  - Query optimiser well tuned;

DBMS and XML

- **General disadvantages**: **twofold mapping for storage and query !**
  - Query navigation is costly, simulated via many joins;
  - Query optimiser looses knowledge on XML nature of the document;
  - Need extra indexes for managing effective *path* queries
- **Disadvantages in the relational case**: (Oracle 8i/9i)
  - Impose a rigid and regular structure via tables;
  - Number of tables is high and much space is wasted;
  - Do exist translation methods but error-prone and DTD is needed.
- **Disadvantages in the OO case**: (Lore at Stanford university)
  - Objects are space expensive, many OO features unused;
  - Management of large objects is costly, hence search is slow.

An XML document is… (W3C project since '96)

- **A simple piece of text containing some mark-up that is self-describing, follows some ground rules and is easily readable by humans and computers.**

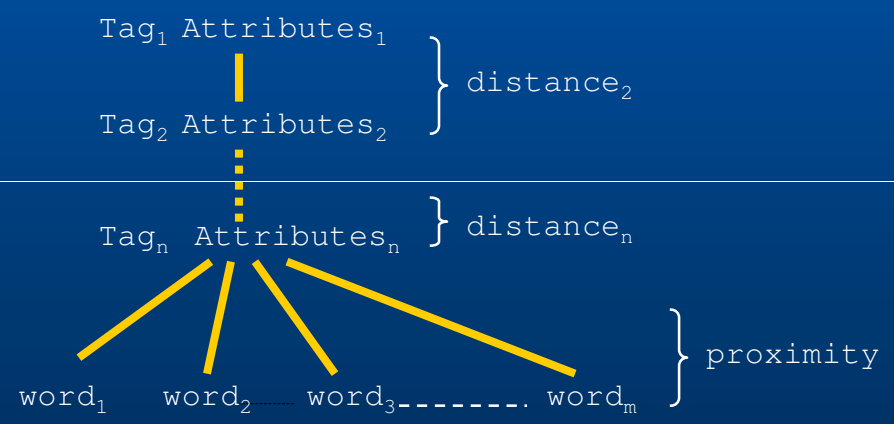
```
<?xml version="1.0" ?>
<weather-report>
    <date> 25/12/2001 </date>
    <time> 09:00 </time>
    <area> Pisa, Italy </area>
    <measurements>
        <skies>  sunny  </skies>
        <temp scale="C"> 2 </temp>
    </measurements>
</weather-report>
```

Tags come in pairs and are possibly nested

Data may be irregular, heterogeneous and/or incomplete

A simple query language

- **Designed to:**
  - **validate the XCDE library;**
  - **search into an Italian literary collection of XML-TEI texts.**
- **Type of queries supported:**



Why this project ?

- **XML is becoming the standard for data representation and exchange amongst applications over the web.**
- **Several XML projects in academia and industry:**
  - **Data models and query languages (XQL, Xquery, Lore,...).**
  - **Tools and applications (Xmill, editors, XSL, EDI,…).**
- **An especially active field:**
  - **Solutions for storing, updating and retrieving information from XML data which may be heterogeneous, irregular and/or incomplete.**

➡ XML offers an opportunity for better Information Retrieval !