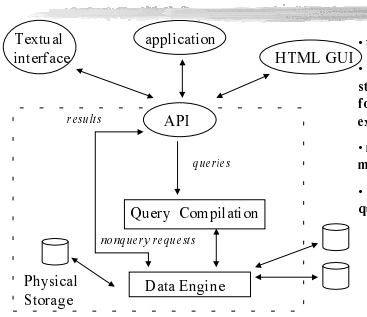
Συστήματα

Εισαγωγή

1. ΜΟΝΤΕΛΑ ΔΕΔΟΜΕΝΩΝ
   - Μοντέλα
   - Γλώσσες Ερωτήσεων
   - Επεξεργασία Ερωτήσεων
   - → **Συστήματα (Lore, Strudel)**
2. ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ
3. ΤΡΟΠΟΙ ΜΕΤΑΔΟΣΗΣ

Lore

Lore Architecture



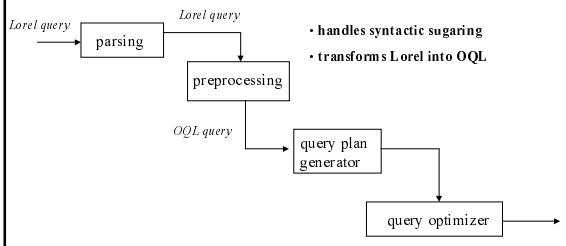
- **tools for browsing results**
- **data guide for seeing the structure of data and formulating queries by example**
- **mechanisms for viewing multimedia types**
- **saving frequently asked queries**

Lore Architecture

- Data can be put into a Lore database through:
  - update statements
  - load files in some standard format (e.g, XML files)
- Objects are arranged in physical disk pages
  - large objects may span several disk pages
  - at load time, objects are clustered on pages in depth-first manner

Lore Architecture

Query Compilation

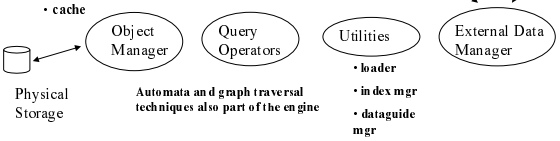


- **handles syntactic sugaring**
- **transforms Lorel into OQL**

Lore Architecture

Data Engine

- **provides the translation layer between OEM data and low-level file data**
- **supports basic primitives such as fetch an object, compare two objects, performs simple coersion, etc**
- **cache**



Automata and graph traversal techniques also part of the engine

Indexes

Two kinds of indexes:

- **Vindex** (value index) takes as **input** a label, a comparator and a value and **returns** all atomic objects having an incoming edge with the specified label and a valye satisying the specified opearator and value (e.g., <ticket, <, 7.50>)
- **Lindex** (link index) takes as **input** an oid *o* and a label *l* and **returns** all (parent) objects having an edge with label *l* going to *o*

  ⇒ backward traversal of the graph

Indexes

- Vindex **useful** for range (inequality) and point (equality) queries - **implemented** as B+-trees
- Lindex **useful** for single object look-ups - **implemented** using linear hash

Query Processing

Example

```
select X
from db.a.b X
where X.c = 5
```

Assume that:

data rooted at db is a tree

10,000 a-elements

each a-element has 500 b-components

only one b-component, say o, has a c-component with value 5

Query Processing

Plan 1 (no indexes) -- *Forward*

visit 10,000 a-elements and for each visit 500 b-components

Plan 2 (with indexes) -- *Backward*

use Vindex (c, =, 5) to get the correct b-component

use two calls of Lindex to verify that there is a path a.b from db

Other Aspects

- The Data Guide

summarize the data in a consize way -- similar to the path index (see 7.4.1)

Other Aspects: Data Guide

Data guide:

finite description of all paths from the root (such paths form a regular expression)

accurate & concise (every path in the data guide occurs only once)

nodes in the data guide corresponds to subsets of nodes in the original graph -- worst case?

Other Aspects

- Managing External Data
- Proximity Search

ranking based on distances in the graph (fuzzy relationships)

Other Aspects

- Views

```
select X
from db.movies X
where X.director = "Hitchcock"
with title, actor, actor.name, actor.picture, script #
```


Other Aspects

- Dynamic OEM and Chorel

subscription queries: e.g., *send me every week the list of new movies with South American directors*

Annotations:

- All vertices: (created date), (deleted date)
- Atomic vertices: (changed-value date old-value)
- Edges between two fixed vertices: (added date) (removed date)

Other Aspects

Extend Lorel to form Chorel (Change-Lorel)

```
select X
from db. <added T> movies X
where X.director.country in db.South-America.country
        and T > "2000/04/04"
```


Other Aspects

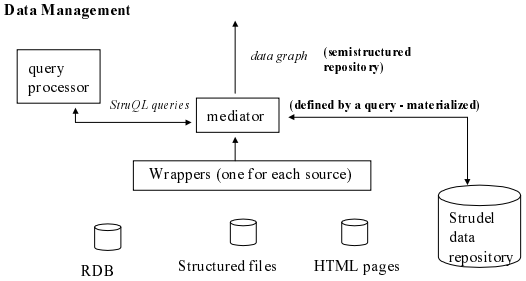
- Ozone

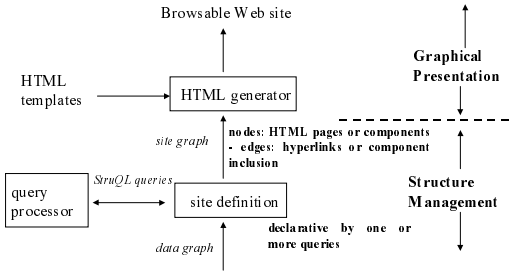
mixing structured and semistructured data

Strudel

Tool for automatic web site construction and maintenance

Data Management





Example

Sources:

- bibtex file (bibtex.bib)
- file of co-authors (in Strudel's syntax) (people.ddl)

Example of Strudel's Model Syntax

people.ddl

```
graph people                          set of nodes
collection Person { }                 with a name
object norman in Person {
        lastname "Ramsey"
        firstname "Norman"
}
object mary in Person {
lastname "Fernandez"
                                      type specification - default
firstname "Mary"                      string
homepage is url "http://www.research.att.com/~mff"
}
```

Data Management

- **wrapper for the bibtex file**

  rewritting

- **data integration**

  using a StruQL query

Data Management

Notes on StruQL query

- $x \rightarrow a \rightarrow y$: edge with label a from node x to y
- where similar to from and where (bind variables)
- create what new nodes to create
- link the new edges
- block structure: the outer block is executed as before

for each binding of the variables, if the where condition in the sublock can be staisfied as well, then the cretae and link statements in the sublock are also executed

Structure Management

Create the site graph

- through a StruQL query
- in general more nodes than HTML pages

Graphical Representation

HTML templates:

HTML files extended with a few Strudel tags

The HTML generator copies from the template into the output HTML file paying attention to certain Strudel tags

Graphical Representation

Strudel tags:

1. SFMT

   takes as an argument a path expression or label

   traverses the corresponding label (or path) and process the subobject's HTML template recursivelly producing another HTML output

   the new HTML text either embedded (default) or stand-alone (if there is a LINK argument)

Graphical Representation

2. SFOR

   An iterator that succesively binds a variable to each subobject reachable by a certain path

3. SIF

   Denotes a conditional statement

Declarative Web Site Management

- define multiple views of the Web site, either by writing a new query or by changing the HTML templates
- clear separation of the three web creation tasks: data, structural and graphical reprsentation
- maintenance
- since the web has an underlying model. Easier to query and transform it in a declarative fashion