Enhancing the Web with DB Technology

Timos Sellis

NTUA

Web and Databases

- For several years researchers from the database systems community have been addressing issues related to data management on the Web.
- Examples
  - XML document management (XQuery, etc)
  - Query processing on the web
- Recent advances related to the Semantic Web as well as the explosion of applications requiring dynamic data extracted from databases call for several new extensions.
- Some novelties “inspired” from data management

New Issues (1)

- The role of **hierarchical schemas** in the Web.
- Hierarchical schemas are used to enrich semantically the available information
  - tree-like structures with syntactic constraints and type information (e.g. DTDs, XML schemas),
  - hierarchies on a category/subcategory basis (e.g. portal catalogs).
- We need a framework to manage hierarchical structures for the Web as first class citizens and not as application layers.

New Issues (2)

- The role of **context** in managing and accessing information.
- Context-dependent data becomes particularly relevant in the Web (e.g. personalization, localization, etc).
- It is important to investigate how to augment the capabilities of information sources so that support for context is part of the data and processing models, not an extra application layer.
- How do we seamlessly introduce context?

New Issues (3)

- The importance of **caching dynamic web objects** in proxies.
- Proxies cache static pages and there has been work on caching dynamic pages
- Nowadays most applications generate dynamic pages with data coming out of database servers
- There is a need for a new kind of proxy that satisfies requests for dynamic web objects, by taking advantage of work in databases in the area of query caching, query rewriting etc.

Rest of talk

- **Handling hierarchical structured data/metadata**
  - (joint work with Theodore Dalamagas)
- Managing Context
- Proxies for handling dynamic data objects

The Semantic Web

- …..the road to the Semantic Web:
  - current Web lacks consistent and strict organization of data
  - difficulties in data sharing and processing in multiple data sources
  - The solution: **Semantic Web**
    - Syntax and semantics in data available on the Web
    - Data has meaning
    - Information is machine-understandable
- Tools: XML* technologies (W3C)

XML* technologies (W3C)

- Syntax and semantics:
  - Data and metadata are marked with tags.
  - XML: the standard encoding format.
  - XML (syntax), RDF (light semantics), OWL (rich semantics)

XML* technologies (W3C)

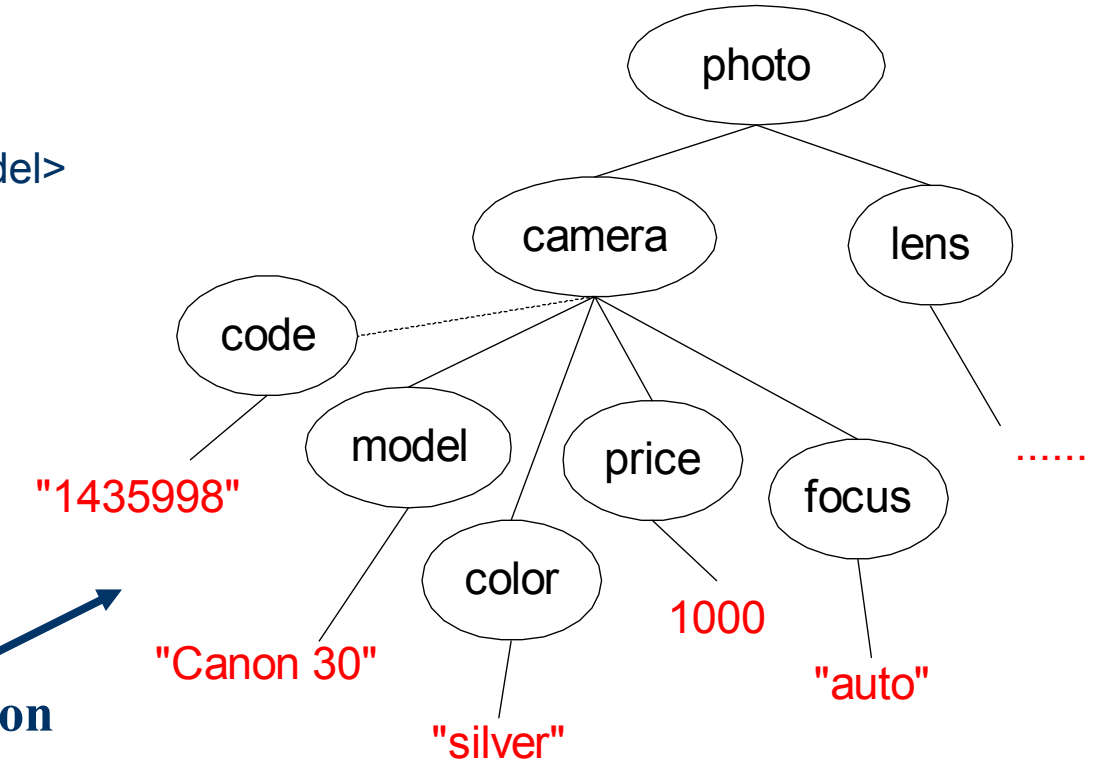
- Our interest: **light semantics**.

Semantic Information

XML* technologies (W3C)

- **Data** marked with tags:

```
<photo>
  <camera code="1435998">
    <model> "Canon 30" </model>
    <color> "silver" </color>
    <price> 1000 </price>
    <focus> "auto" </focus>
  </camera>
  <lens>
    ....
  </lens>
</photo
```



Tree-like representation

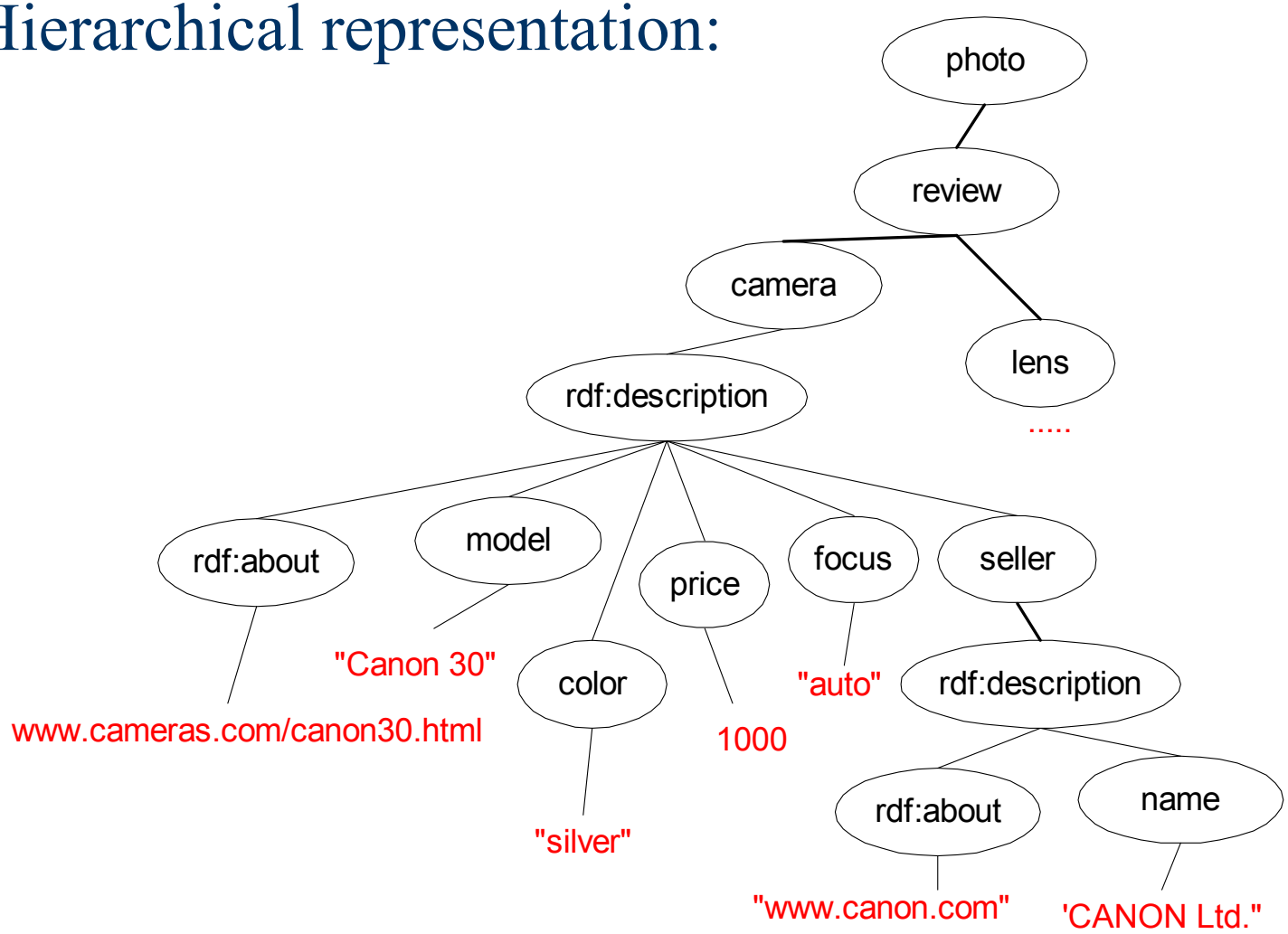
XML* technologies (W3C)

- **Metadata** marked with tags:

```
<photo><review><camera>
<rdf:description rdf:about="www.cameras.com/canon30.html”>
    <model> ”Canon 30” </model>
    <color> ”silver” </color>
    <price> 1000 </price>
    <focus> ”auto” </focus>
    <seller>
        <rdf:description rdf:about=”www.canon.com”>
        <name> “CANON Ltd.” </name>
        </rdf:description>
    <seller>
</rdf:description>
</camera><lens> … </lens></review></photo>
```

XML* technologies (W3C)

- Hierarchical representation:



The role of hierarchies

- We consider hierarchical structures (**hierarchies** from now on) as an important tool to support the development of the Semantic Web.
  - XML: tree-like structures (ignoring IDREFs)
  - RDF(s): graph structures
- We are interested in tree-like hierarchical structures
  - XML/RDF encodings

The problem

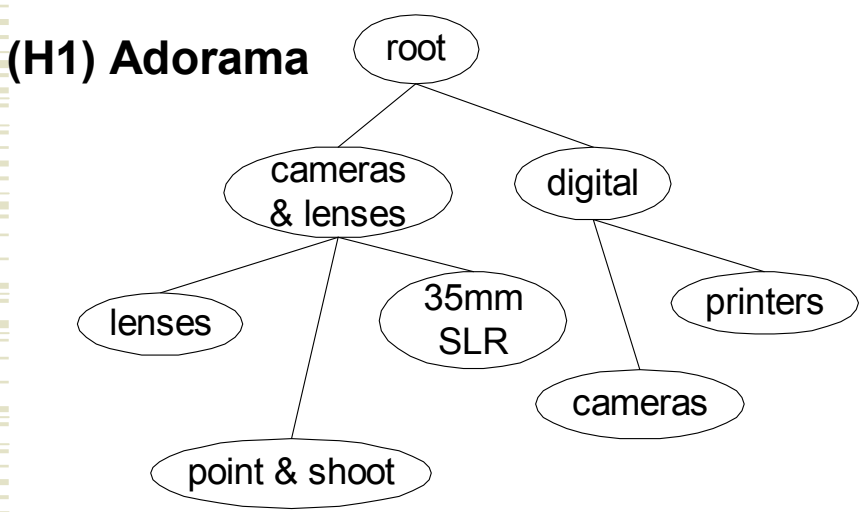
- Hierarchies are nowadays treated as sets of individual elements (i.e. nodes)
- Hierarchies = simple semantic guides for
  - browsing
  - Posing path expression queries:
    /cameras/manual/item[price<1000]

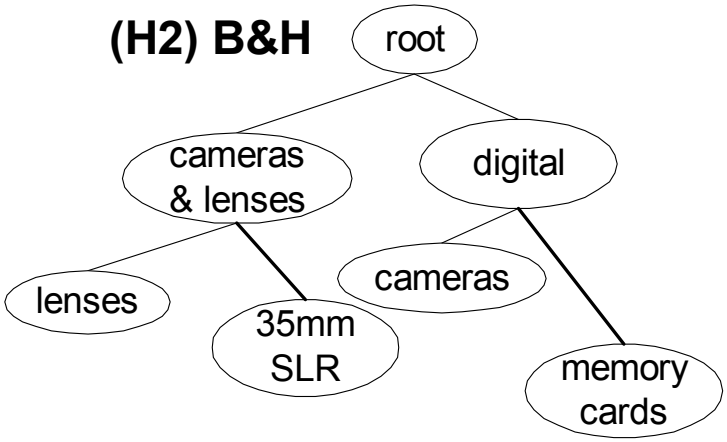
The problem

- There are many hierarchies on the Web that organize data for a given knowledge domain.
- New type of queries need to be supported:
  - 'find hierarchies that organize photographic equipment **similarly** to a given hierarchy' (structural/semantic similarity).
  - 'find the part of a hierarchy which is not present in another hierarchy' (manipulation of structural information).

The problem

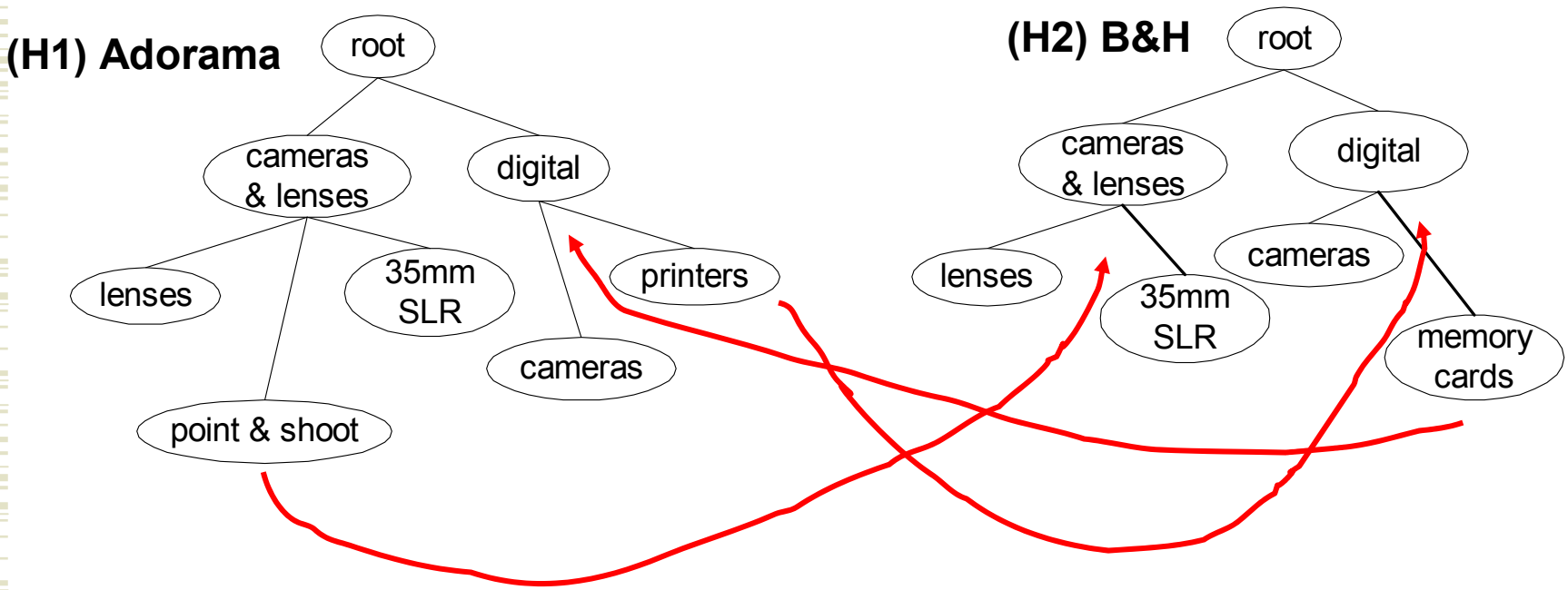
- Structural/Semantic Similarity





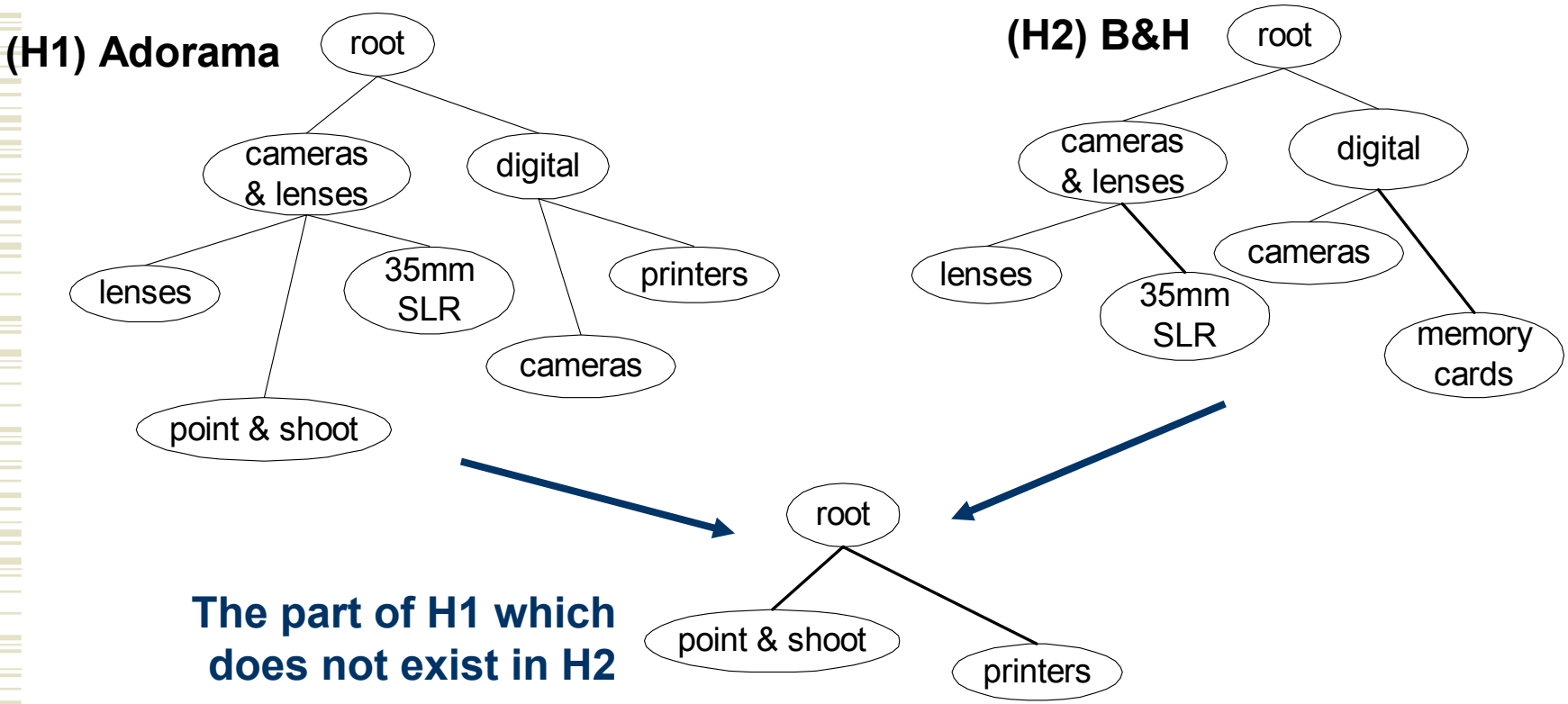
The problem

- Structural/Semantic Similarity



The problem

- Manipulation of structural information



Major contributions

- Upgrade hierarchies to **first-class citizens.**
- Set up a framework to manipulate hierarchies:
  - **Algorithms to detect homologous hierarchies**.
  - **Manipulate structural information in multiple hierarchies.**
  - **Manipulate hierarchies and data organized in a uniform way – tree structured relations**.

Research issues we consider

- A methodology to detect homologous hierarchies.
  - Define **distance metrics** to capture the structural similarity among hierarchies, and design algorithms to calculate them.
  - Apply **clustering algorithms** to detect groups of structurally similar hierarchies.

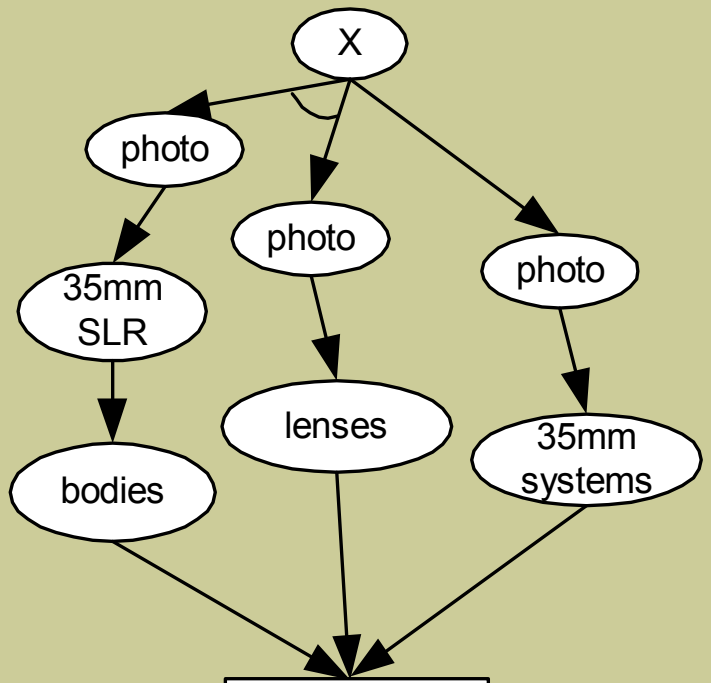
Research issues we consider

- Structural manipulation of hierarchies.
  - Study the **algebraic properties** of hierarchies as tree-like structures.
  - Define three **operators** to manipulate their structural information (union, intersection, difference), having similar properties to those of set theory.

- Manipulating data and hierarchies.
  - Define operators that **combine**
    - Manipulation of paths in hierarchies and
    - Traditional relational queries on data.

Find cameras giving their model and the corresponding lens

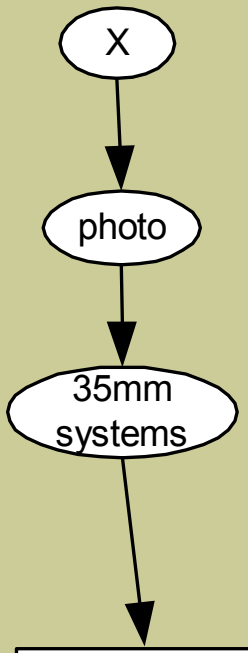
$\pi_{\langle model,\ lens\_id \rangle\ \langle \#2 \rangle}$(SLR systems)



(a)

SLR systems

brand	model	price	lens_id
Canon	EOS-3	990	1
Nikon	N65	205	2
Pentax	ZX-M	148.5	2
...	...	...	...



(b)

SLR systems

model	lens_id
EOS-3	1
N65	2
ZX-M	2
... ...	...

Rest of talk

- Handling hierarchical structured data/metadata
- **Managing Context**
  (joint work with Yannis Stavrakas)
- Proxies for handling dynamic data objects

Context

- Context is a **tool** for reasoning with viewpoints and background beliefs, and a **mechanism** for dealing with complexity, heterogeneity, and partial knowledge.
- For the **user**, context (*query context*) expresses:
  - The preferences, the viewpoint, the implicit assumptions used to **interpret** data...
  - ...but also the capabilities of a device (cell, PDA, laptop).
- For the information **provider** (*data context*):
  - Management of **variants** of the same information that address different groups of users.
- For the information management **systems**:
  - Abstraction mechanism (*viewpoint abstraction*).
  - Allows to **focus** on some views of the reality, ignoring others.

Our approach

- The pivotal question:
  - **How to incorporate context in the Web as a first-class citizen?**
- In our approach:
  - Every information entity presents different **facets** that hold under different **worlds**.
  - Every facet is related to a context, which represents a **set** of possible worlds.
  - Each world corresponds to an interpretation frame of the information receiver, under which data obtain **substance**.
- Context is expressed through **context specifiers**.

Context specifiers

- A **world** is defined by assigning a value to every dimension in a set of dimensions **D**:

```
lang=greek, detail=low, format=pdf
```

- A **context specifier** represents the set of worlds that conform to given constraints:

```
[lang=greek, detail in {low,medium}]
[time in {8..13,17..20}]
[detail=high, lang in {en,gr} | format=pdf]
```

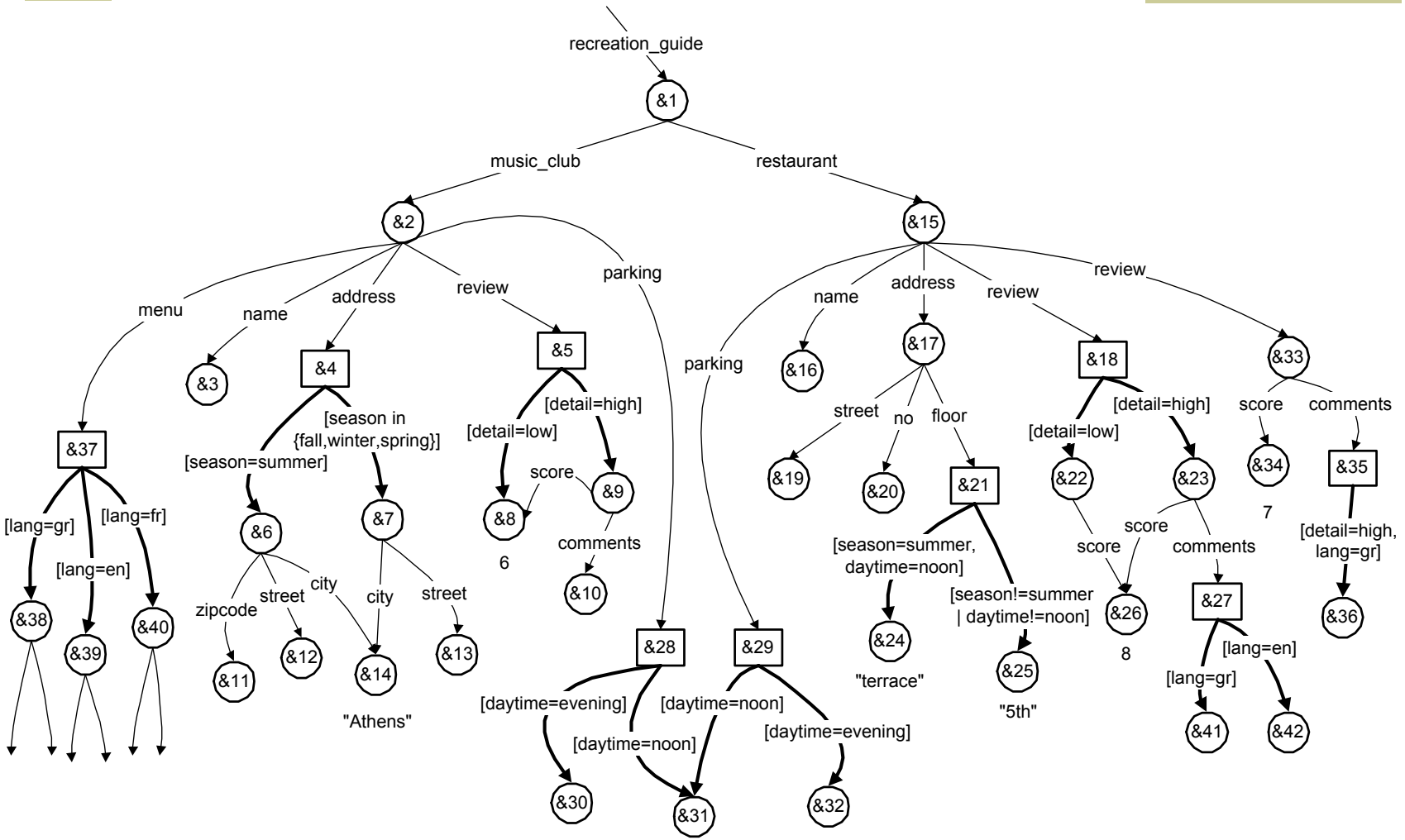
- **Context operations**, maintain the correspondence with the relevant sets of worlds.

$$W_D(c_1 \cap^c c_2) = W_D(c_1) \cap W_D(c_2)$$

How to incorporate context?

- Start with SSD: simple and expressive model (OEM).
  - Incorporate context as first-class citizen.
  - **MOEM**: conglomeration of OEM variants that hold under different worlds.
  - **MQL**: context used in queries, based on Lorel.
- Also **MXML**, **MDTD**: context + XML.
- Any benefits?
  - **Reduction**, a uniform and flexible mechanism for tailoring information to different frames of interpretation.
  - **Management** of information according to the context under which it holds.
  - Capability to formulate **cross-world** queries.

MOEM: a recreation guide



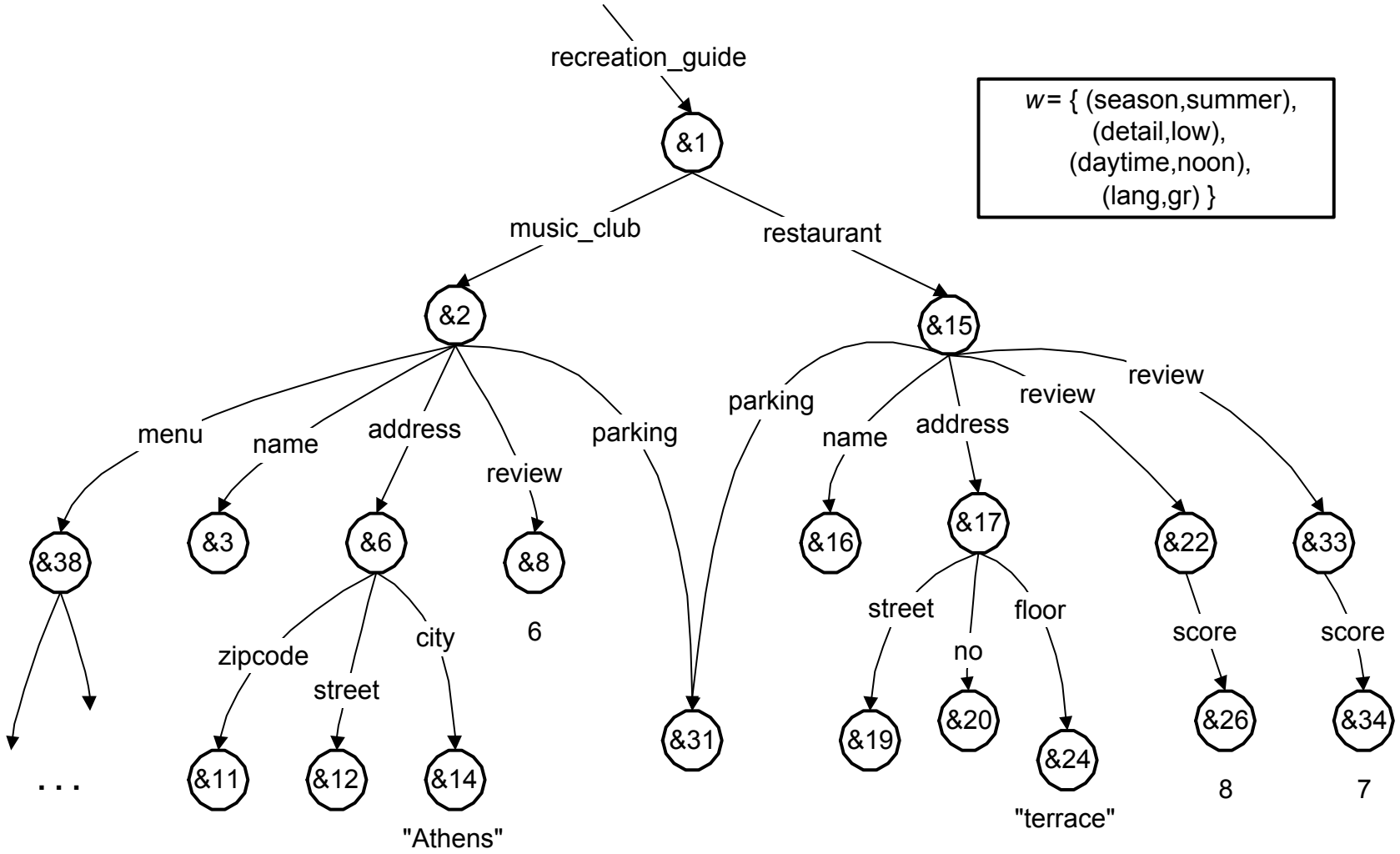
The interesting issues

- **Explicit context**: appears on labels of MOEM edges.
  - It has meaning **only** within the boundaries of a single multidimensional entity.
- **Inherited coverage** of a node or an edge:
  - An object holds only under the worlds that some "father" node holds.
  - An object holds only under the worlds it has access to some atomic node (leaf).
  - *A node or an edge* ***holds*** *under a world w* ***if w belongs to the corresponding inherited coverage***.
- **Path inherited coverage**.

...and interesting results

- Process "**reduction to OEM**" under the world *w*:
  - Extracts from an MOEM the OEM facet that holds under *w*
- Process "**partial reduction**" for a context specifier *c*:
  - Extracts a graph that exactly incorporates **all OEM facets** that hold under the worlds in *c*.
- **Canonical form** of an MOEM:
  - Reduced to the **same** OEMs as the original MOEM.
  - Avoids unnecessary dependencies which lead to update, insert, and delete **anomalies**.
  - ***Used to formulate and evaluate queries***.

Reduction to OEM



MQL

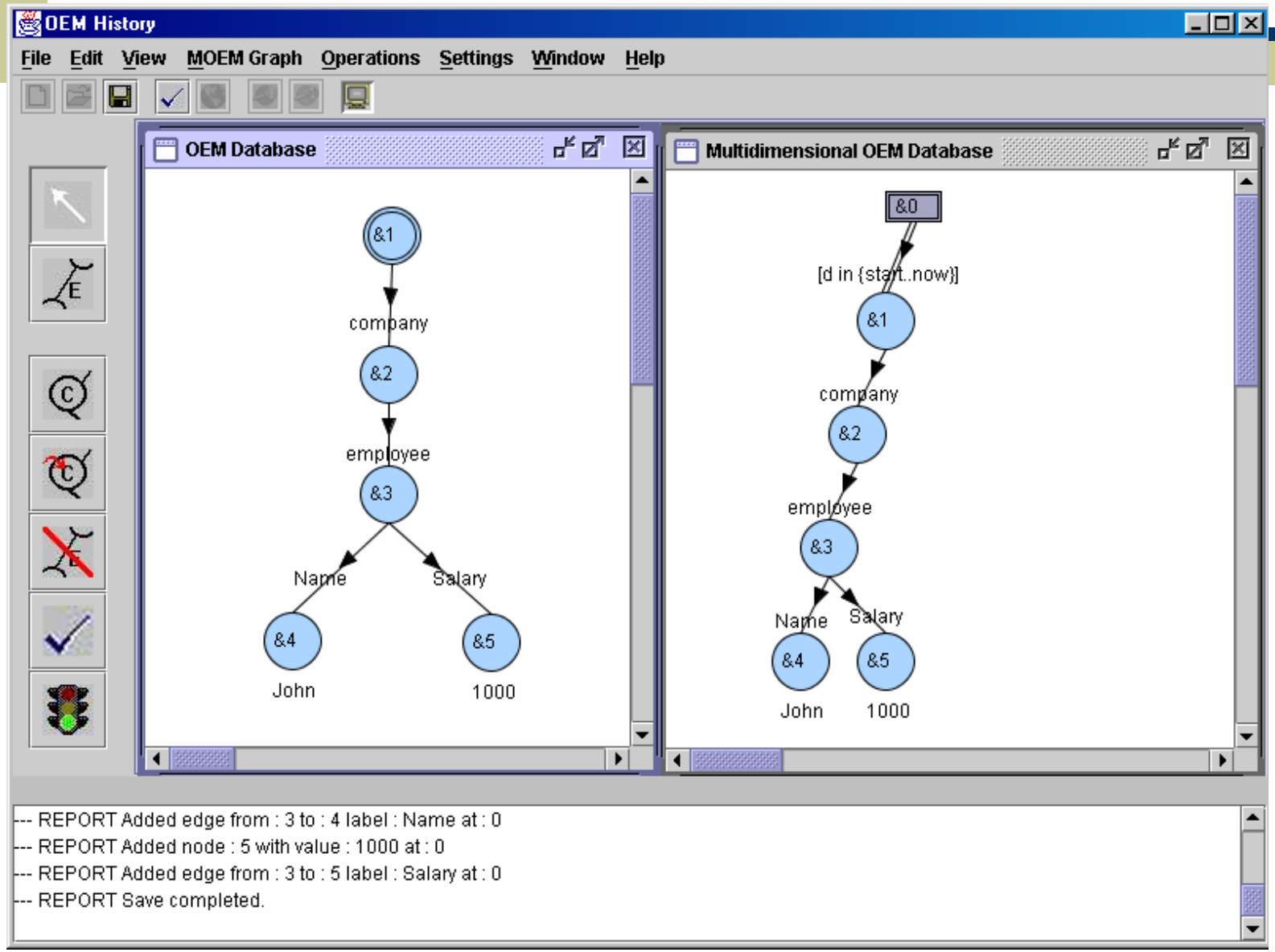
- **Context path expressions**: context + path expressions.
  - **Semantics**: joins between OEMs in an MOEM that hold under different worlds.
- **Cross-world** queries, *take advantage of the grouping of facets in an MOEM.*

```
select winter_floor: Y
from recreation_guide.restaurant X,
     X.[season=winter]address.floor Y,
     X.[season=summer,daytime=noon]address.floor Z,
where Z="terrace"
```

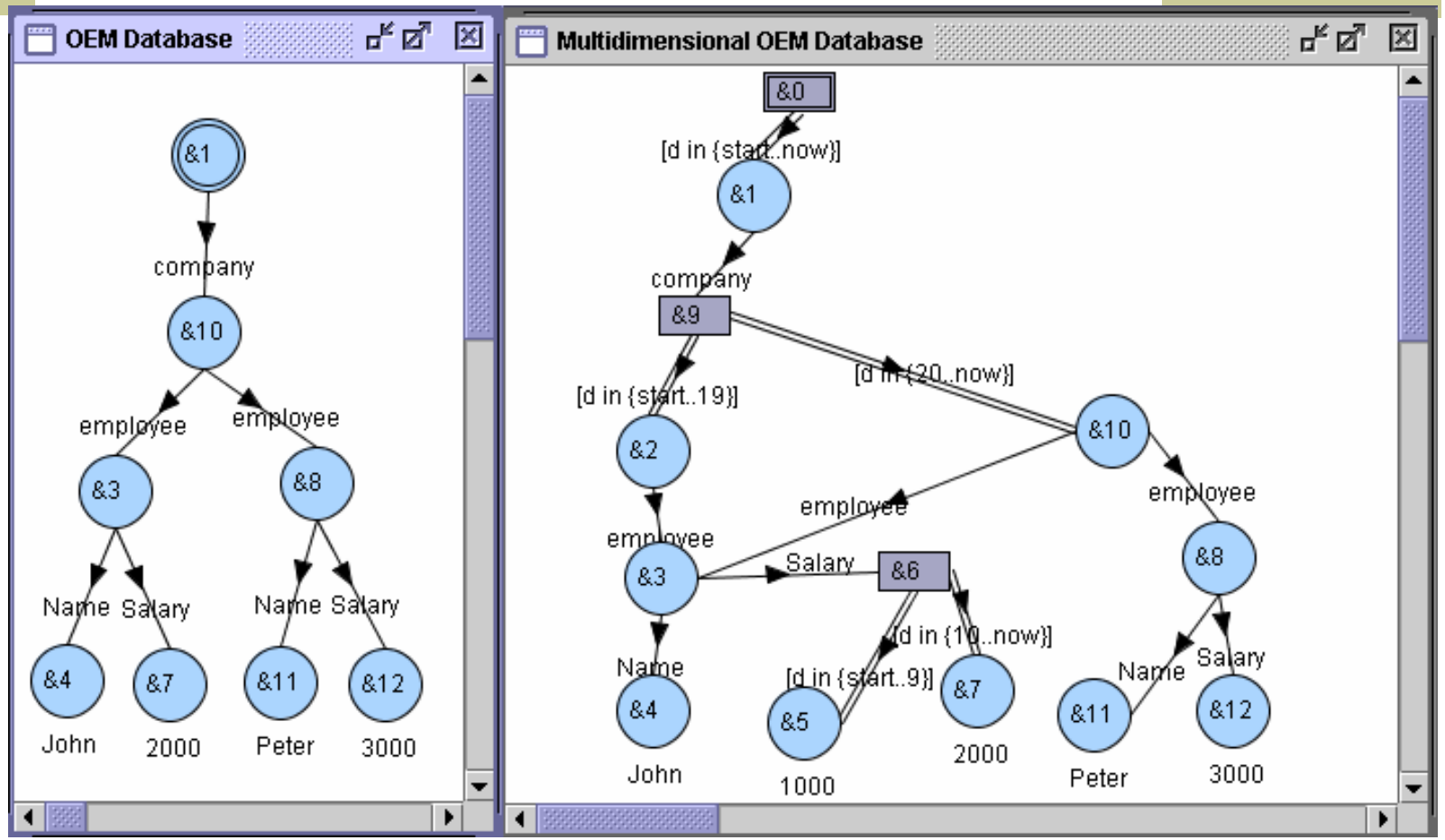
- **Context variables**, and the "`within`" clause:

```
select compr_menu: {menu: Y, context: [X]}
from recreation_guide.music_club.menu::[X] Y
within [X] * [lang in {gr,en}] != [-]
```

An application: handling history



OEM History: after a few changes



Querying OEM histories with MQL

- Examples:
  - Find Peter's salary at **time instance** 32.

```
select salary: S
from [d=32]db.company.employee{X}.salary S
where X.name = "Peter"
```

  - Find the employees whose salary **has not changed** since time instance 32.

```
select employee: {name: Z, salary: S}
from db.company.employee{X}.salary::[Y][-] S,
     X.name Z
within [Y] >= [d in {32..now}]
```

- **Interesting**: possible to represent and query the history of an *M*OEM **with no extra concepts**.

MXML

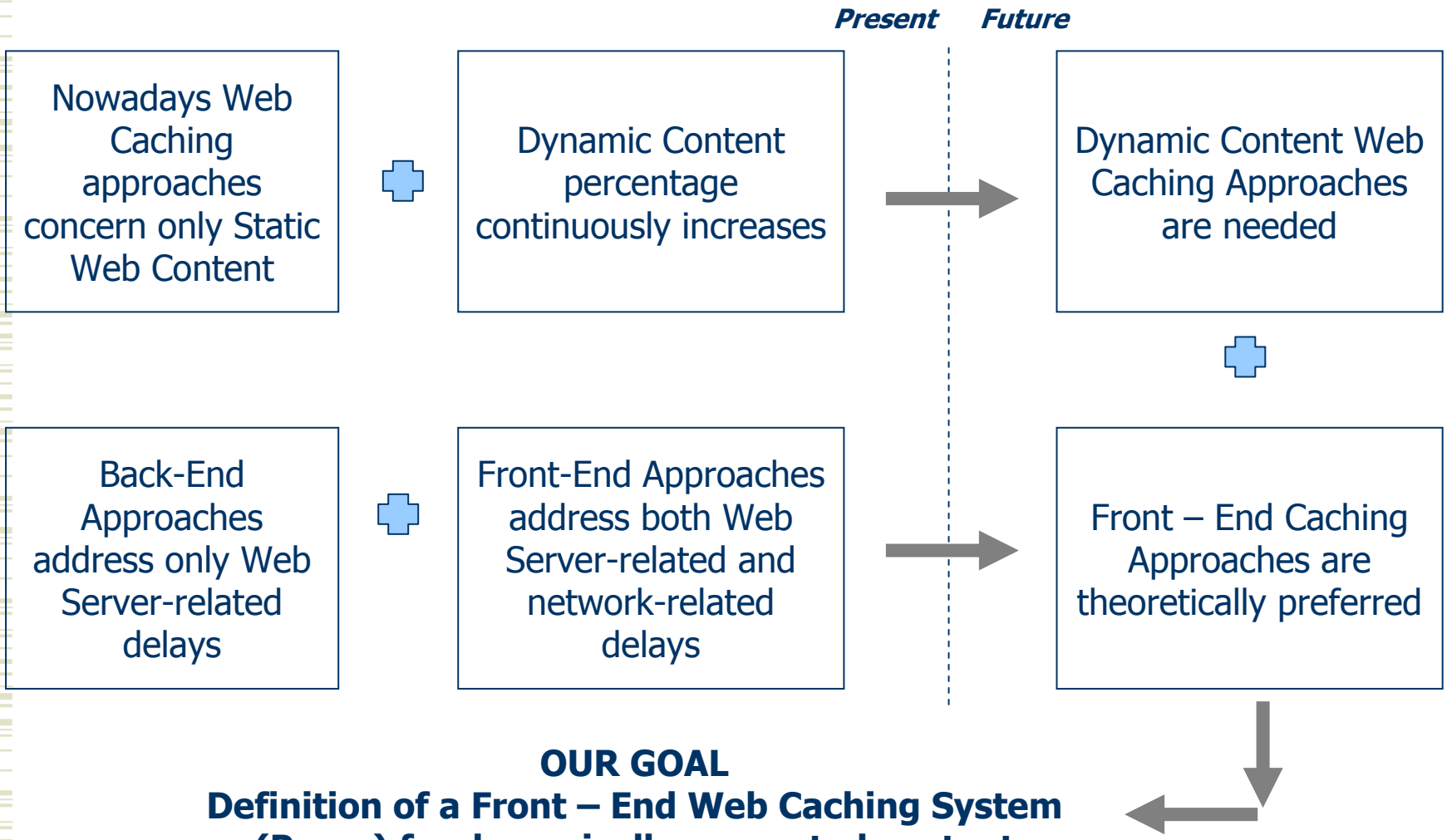
```
<menu>
   <salad vegetarian = [season = summer] "yes" [/]
                       [default] "no" [/]>
      <name> Chef's salad </name>
      <@comment>
         [language = English, detail = low]
            <comment> A traditional salad. </comment>
         [/]
         [language = English, detail = high]
            <comment> A salad which ... </comment>
         [/]
         [language = French, detail in {low, high}]
            <comment> Une salade traditionelle.
   </comment>
         [/]
      </@comment>
...
```

just part of an MXML document...

Rest of talk

- Handling hierarchical structured data/metadata
- Managing Context
- **Proxies for handling dynamic data objects**
  (joint work with Manolis Veliskakis)

Motivation


Present
Future
Nowadays Web Caching approaches concern only Static Web Content
Dynamic Content percentage continuously increases
Dynamic Content Web Caching Approaches are needed
Back-End Approaches address only Web Server-related delays
Front-End Approaches address both Web Server-related and network-related delays
Front – End Caching Approaches are theoretically preferred
OUR GOAL
Definition of a Front – End Web Caching System (Proxy) for dynamically generated content

Static vs. Dynamic Content (1)

Static Content

- already **stored** on the Web Server
- **low** update rate
- presentation and content are **request-independent**
- one request corresponds to **one** Web Page

Dynamic Content

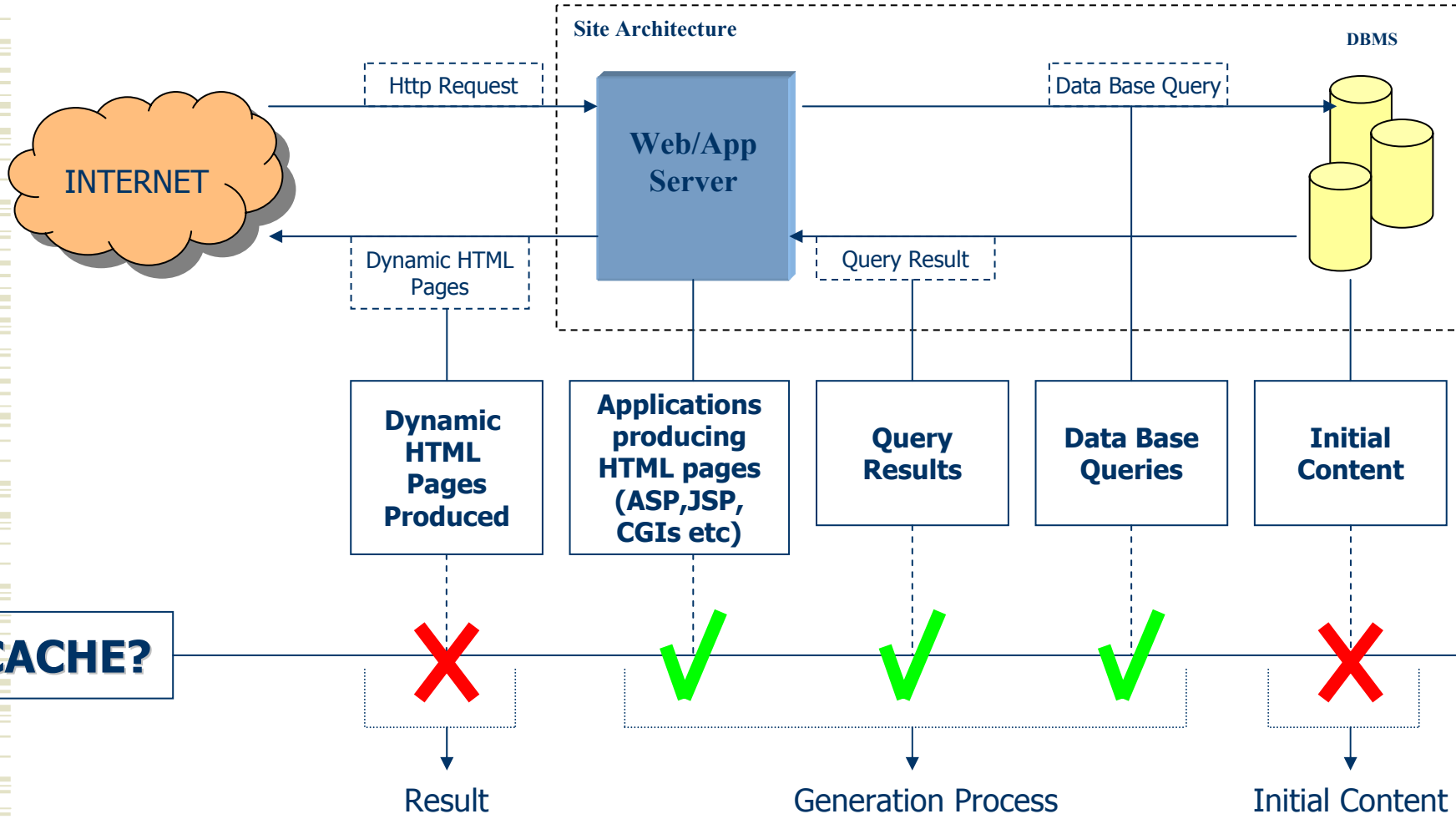
- produced **"on demand"**
- **high** update rate
- presentation and content are **request-dependent**
- one request corresponds to **several** Web Pages

Static vs. Dynamic Content (2)

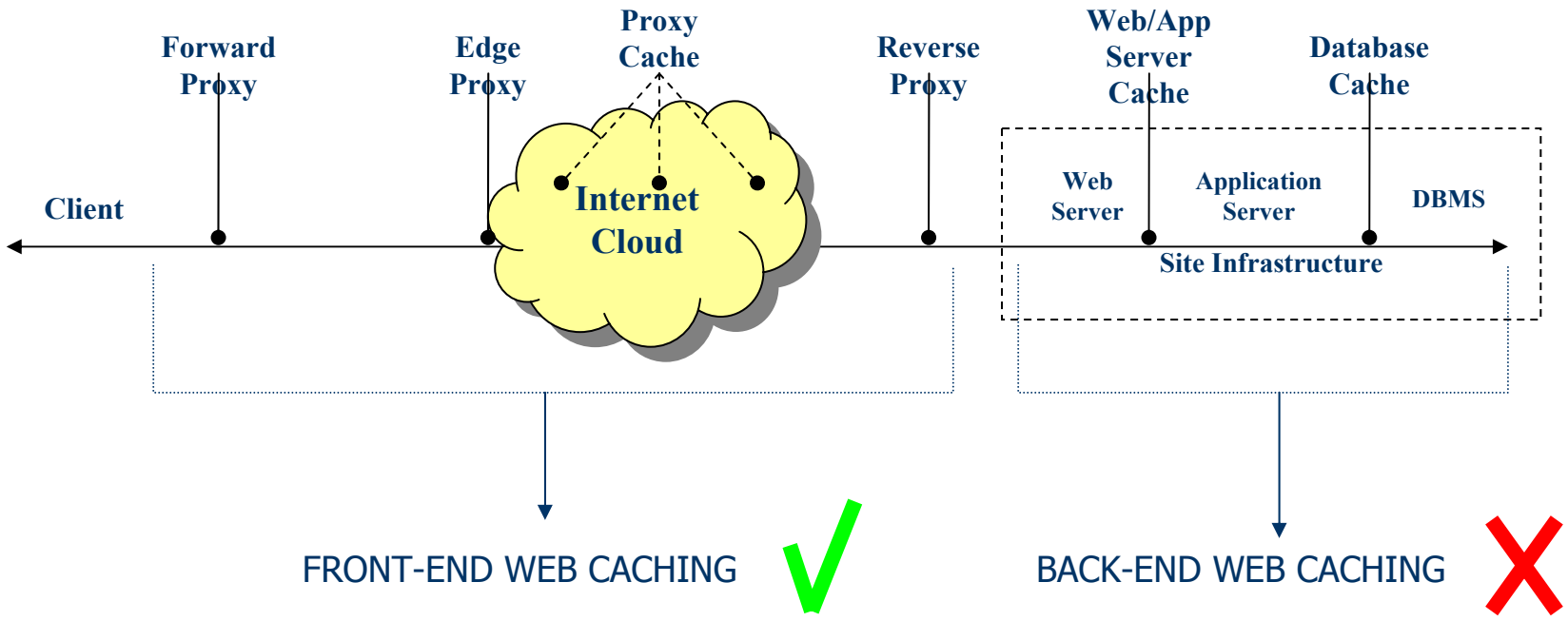
Observations

- Web Caching approaches concerning Static Content are not appropriate for Dynamic Content
- The following issues must be re-addressed and solved
  - **What**, **where**, and **how** to cache?
  - **How** to use the cache?
  - **Replacement** Policy
  - Cache **Consistency**

What to Cache?



Cache Deployment



CHALLENGE:

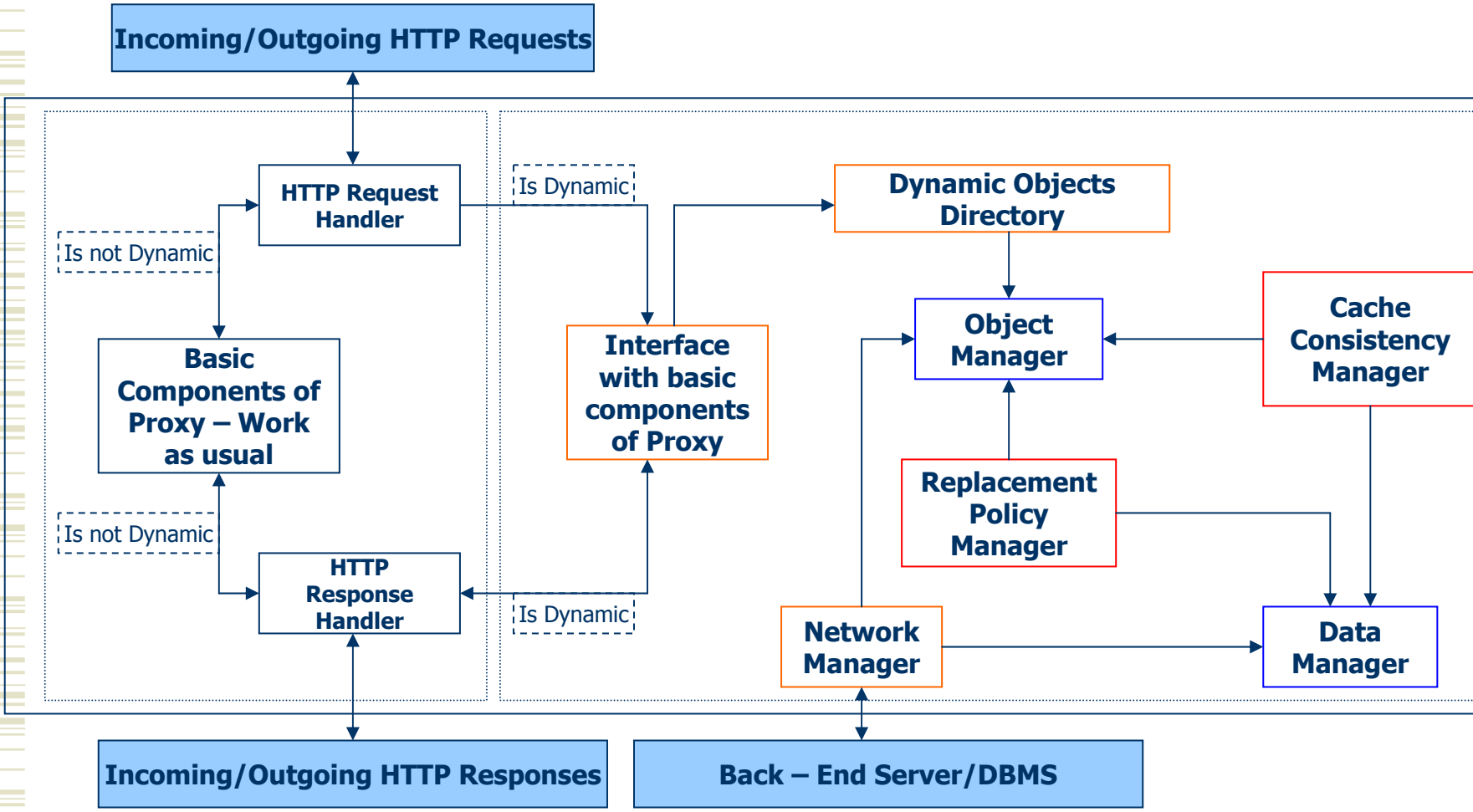
HOW TO GIVE APPLICATION AND DATABASE LOGIC TO A PROXY OUTSIDE THE SITE INFRASTUCTURE

Incorporating Application and Database Logic to Proxies

General Characteristics

- Attach every dynamically generated HTML page to its corresponding Application (e.g Cache Applets)
- Attach every dynamically generated HTML page to its corresponding back-end Content
- Cache both corresponding Applications and back-end Content in the Proxy
- Give Proxy the ability to produce on request the dynamic HTML Pages
- Do all the above as transparently as possible

Proxy Structure

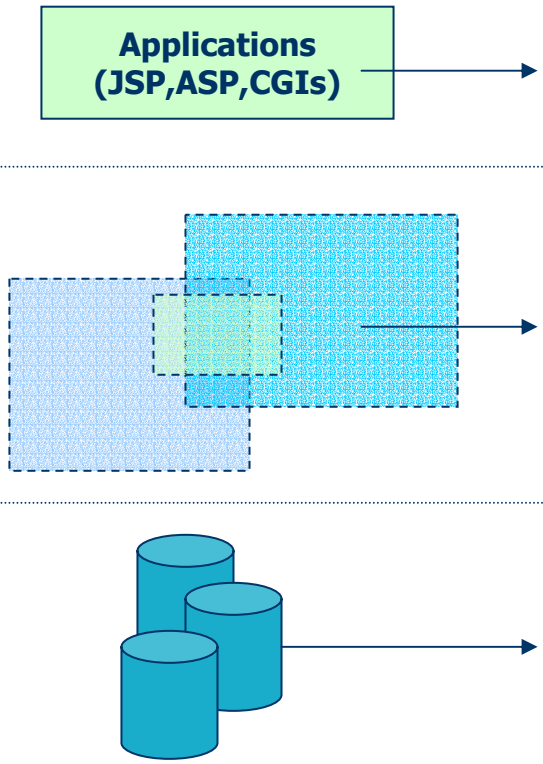


Proxy Structure

MAIN COMPONENTS

- **Dynamic Objects Directory:** Mapping HTTP Requests/Responses to cached Applications (JSP,ASP,CGIs)
- **Object Manager:** Manipulates the cached Applications
- **Replacement Policy Manager:** Defines the Replacement Policy of cached Applications and Content
- **Cache Consistency Manager:** Defines the Consistency Policy of cached Applications and Content
- **Data Manager:** Stores and manipulates the cached content (minimal DB requirements)

3-Layer Replacement Policy

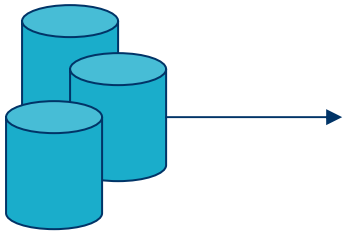


Open Issues

- One type of Replacement Policy for all types of cached objects?
  - Works independently for each type of cached objects ?
- Different types of Replacement Policies for each type of cached objects?
  - If an application is removed what happens with its corresponding cached content?
- How does the dynamic nature of cached web objects affect the type of the Replacement Policy (Weight-Based, LRU, LFU etc)?

2-Layer Cache Consistency

LAYER No2 :
Presentation Level Consistency



LAYER No1 :
Content Level Consistency

Open Issues

- One type of Cache Consistency Policy for both layers?
- Definition of the relationship between Presentation and Content Level
- How does the dynamic nature of cached web objects affect the type of the Cache Consistency Policy (Invalidation, Validation etc)?

Summary

- Information management in the Web era requires the development of many novel ideas.
- Attention is needed for
  - **modeling** richer information structures (e.g. trees/catalogs/…)
  - **incorporating** as much as possible information-related data and/or metadata at the proper place, i.e. with the information itself (e.g. context)
  - making **processing** effective and efficient
- An interesting road with a lot of opportunities