Object-Oriented Databases
Support for Context-Aware Data Management

• Version Model
• Query Processor
• Implementation
It's been a long way...

Mainframe Computer  Workstation  Personal Computer  Laptop Computer

Palmtop Computer  Media Phones  Disappearing Computer

What will be next?
...and the road goes on and on.

- So far, information systems have always coped with these ever-changing platforms and requirements
  - hierarchical databases
  - network databases
  - relational databases
  - object-oriented databases
  - object-relational databases
  - engineering databases
  - lightweight databases
  - personal databases
  - mobile databases
  - context-aware databases?
The Need for Context-Aware Computing

- **Mobile computing**
  - device limitations, such as reduced interaction bandwidth
  - location, environment, tasks, preferences, history, device characteristics, ...

- **Pervasive computing**
  - lack of traditional interfaces, such as keyboards or screens
  - environment, tasks, moods, preferences, history, personality, background, ...

- **Web engineering**
  - content adaptation and proactive behaviour
  - personalisation, internationalisation, access channel or mode, ...
Solutions for Context-Aware Computing

- Models
  - context representation
  - context management

- Infrastructures
  - context gathering
  - context processing and augmentation
  - trigger-based application adaptation

- CASE tools
  - model-based generation of context-aware applications
Solutions for Context-Aware Computing

- Very few context-aware information systems, but...
  - temporal databases
  - engineering databases (CAD, CAM)
  - software configuration management

...have addressed comparable problems in the past

- Upshot
  - stratum approaches built on top of existing systems do not work
  - experience in models, storage, indexing and query languages
  - context-aware data management has different requirements in terms indexing and query processing
## Positioning of Our Work

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requirements</strong></td>
<td></td>
</tr>
<tr>
<td>Use cases</td>
<td>UML use case diagrams, ...</td>
</tr>
<tr>
<td>User studies</td>
<td>Usability, field tests, ...</td>
</tr>
<tr>
<td>Specifications</td>
<td>Mock-ups, descriptions, ...</td>
</tr>
<tr>
<td><strong>Modelling</strong></td>
<td></td>
</tr>
<tr>
<td>Data Models</td>
<td>E/R, Relational Model, OM, ...</td>
</tr>
<tr>
<td>Software Models</td>
<td>OMT, UML class diagrams, ...</td>
</tr>
<tr>
<td>Domain-specific Models</td>
<td>OOHDM, WebML, Hera, ...</td>
</tr>
<tr>
<td><strong>Implementation</strong></td>
<td></td>
</tr>
<tr>
<td>Manual</td>
<td>IDE (Eclipse, Visual Studio, ...)</td>
</tr>
<tr>
<td>Semi-Automatic</td>
<td></td>
</tr>
<tr>
<td>Automatic Code Generation</td>
<td>WebRatio</td>
</tr>
<tr>
<td><strong>Platforms</strong></td>
<td></td>
</tr>
<tr>
<td>Relational Databases</td>
<td>SQL Server, mySQL, Oracle, ...</td>
</tr>
<tr>
<td>Object-Oriented Databases</td>
<td>ObjectStore, db4o, OMS, ...</td>
</tr>
<tr>
<td>(Programming) Languages</td>
<td>Java, C#, JSP, XML, XSLT, ...</td>
</tr>
</tbody>
</table>
Context-Aware Information Systems

- **Goals**
  - Context-dependent query processing
    - mobile computing, pervasive computing, web engineering
  - Support for application development
    - web engineering, software engineering, product engineering

- **Approach**
  - Context notion and representation
  - Two-dimensional version model
    - Alternative versions (variants) for run-time context-awareness
    - Revisional versions (revisions) for design-time system evolution
  - Matching algorithm to select content-dependent variants
    - Compute best match rather than exact match
  - Integration into an object-oriented data management system
Context in Information Systems

- Context information is optional
- Information system has a well-defined default behaviour
- Available context information may vary
- Result is augmented or improved rather than specified by context
- Context representation needs to be general and open
Context, Context Space and Context State

- **Context space**
  - context dimensions that are relevant to an application
  - $S = \{name_1, name_2, ..., name_n\}$
    - $\forall i: 1 \leq i \leq n \Rightarrow name_i \in$ NAMES and therefore $S \subseteq$ NAMES

- **Context Value**
  - $c = <name, value>$, where $name \in$ NAMES and $value \in$ VALUES

- **Context**
  - $C(S) = \{<name_1, value_1>, <name_2, value_2>, ..., <name_m, value_m>\}$
    - $= \{c_1, c_2, ..., c_m\}$
    - $\forall i: 1 \leq i \leq m \Rightarrow name_i \in S$ and $\forall c_i, c_j \in C: c_i = c_j \Rightarrow name_i = name_j$

- **Context state**
  - special context $C_\star(S)$, $\forall name \in S$: $\exists <name, value> \in C_\star(S)$
Revisions and Variants

Revisions

- design-time
- "off-line" queries
- targeted at developers
- implicit and explicit creation
- serial

Variants

- run-time
- "on-line" queries
- targeted at users
- explicit creation only
- parallel
OM Object Metamodel
Version Model

- Metamodel extended with variants and revisions
  - variants are structurally favoured over revisions
  - run-time performance versus design-time performance
- Variants define a variant context $C_v(S)$
  - defines in which context a variant can be used
  - no two variants of an object can define the same $C_v(S)$
- Revisions are identified by a revision number
  - ascending sequence
  - counter is incremented when a new version is generated
- All versions of an object have the same types
Versioned OM Object Metamodel
Evolution of a Versioned OM Object
Identifying and Referencing Objects

- Both specific and generic references are supported
  - Object identifier format:
    - "id" @ "version" [ "variant" ]
  - Concept of **default variant** and **latest version**
  - Partially specified references can be completed automatically

- **Versioning of object graphs**
  - Based on object references
    - both generic and specific references can be used
    - local, managed within objects, uni-directional
    - versioning of relationships is dependent on versioning of objects
  - Based on associations
    - relation between objects based on tuples of object identifiers
    - represented as object with revisions and variants
    - global, managed outside objects, bi-directional
    - versioning of relationships is independent of versioning of objects
Automatic Completion of Object Identifiers

- For each dimension of the version model there is a default representation
  - Latest version
  - Default variant

- Partially specified references are completed as follows
  - 927[2]  ➤ 927@4[2]  \textit{default version}
  - 927@3  ➤ 927@3[0]  \textit{default variant}
  - 927  ➤ 927@4[0]  \textit{default version and variant}
Context-Aware Query Processing

MATCH(o, C*(S))
1 V₀ ← rng(HasVariants dr({o}))
2 V₁ ← V₀ × (x → (x × rng(HasProperty dr({x}))))
3 V₂ ← V₁ × (x → (dom(x) × fₛ(C*(S), rng(x))))
4 s₁ ← max(rng(V₂))
5 V₃ ← V₂ % (x → rng(x) = s₁)
6 if |V₃| = 1 ∧ s₁ ≥ s_min
   then v ← V₃ nth 1
   else v ← rng(DefaultValue dr({o})) nth 1
9 return v

Simple Scoring Function

\[ fₛ(C₁, C₂) = \frac{1}{|N₁|} \sum_{n \in N₁} f_i(n, C₁, C₂) \]

\[ f_i(n, C₁, C₂) = \begin{cases} 
1 & \exists c₁ \in C₁, c₂ \in C₂ : name₁ = name₂ = n \land value₁ \equiv value₂ \\
0 & \text{otherwise.}
\end{cases} \]

Invoked for every object o accessed by the query evaluator

- Scoring function \( fₛ \) assigns a score value to every variant \( v \) of \( o \)
- Indicator function \( f_i \) uses matching condition \( (\equiv) \) for context values
- The variant with the highest score \( s_{max} \) is returned, if
  - there is only one variant with \( s_{max} \)
  - the score \( s_{max} \) is above the system threshold \( s_{min} \)
Syntax for Context and Property Values

The indicator function \( f_i \) supports the following syntax for both context and property values:

<table>
<thead>
<tr>
<th>Type</th>
<th>Syntax</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>atom</td>
<td>( x )</td>
<td>Atomic value</td>
<td>en, 27</td>
</tr>
<tr>
<td>set</td>
<td>( x_1{:}x_n )</td>
<td>Set of atomic values ( S := {x_1, \ldots, x_n} )</td>
<td>at:ch:de, red:blue</td>
</tr>
<tr>
<td>range</td>
<td>( x_{\text{min}}..x_{\text{max}} )</td>
<td>Range of atomic values ( I := [x_{\text{min}}, x_{\text{max}}] )</td>
<td>5.5..7.0, a..f</td>
</tr>
<tr>
<td>star</td>
<td>*</td>
<td>Wildcard</td>
<td>*</td>
</tr>
</tbody>
</table>
Matching Condition ($\equiv$) for Context Values

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>Matching Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATOM</td>
<td>ATOM</td>
<td>$x = y$</td>
</tr>
<tr>
<td>ATOM</td>
<td>SET</td>
<td>$x \in y$</td>
</tr>
<tr>
<td>ATOM</td>
<td>RANGE</td>
<td>$y_{\min} \leq x \leq y_{\max}$</td>
</tr>
<tr>
<td>ATOM</td>
<td>STAR</td>
<td>$T$</td>
</tr>
<tr>
<td>SET</td>
<td>ATOM</td>
<td>$y \in x$</td>
</tr>
<tr>
<td>SET</td>
<td>SET</td>
<td>$x \cap y \neq \emptyset$</td>
</tr>
<tr>
<td>SET</td>
<td>RANGE</td>
<td>$\exists k \in x : y_{\min} \leq k \leq y_{\max}$</td>
</tr>
<tr>
<td>SET</td>
<td>STAR</td>
<td>$T$</td>
</tr>
<tr>
<td>RANGE</td>
<td>ATOM</td>
<td>$x_{\min} \leq y \leq x_{\max}$</td>
</tr>
<tr>
<td>RANGE</td>
<td>SET</td>
<td>$\exists k \in y : x_{\min} \leq k \leq x_{\max}$</td>
</tr>
<tr>
<td>RANGE</td>
<td>RANGE</td>
<td>$\max(x_{\min}, y_{\min}) &lt; \min(x_{\max}, y_{\max})$</td>
</tr>
<tr>
<td>RANGE</td>
<td>STAR</td>
<td>$T$</td>
</tr>
<tr>
<td>STAR</td>
<td>ATOM</td>
<td>$T$</td>
</tr>
<tr>
<td>STAR</td>
<td>SET</td>
<td>$T$</td>
</tr>
<tr>
<td>STAR</td>
<td>RANGE</td>
<td>$T$</td>
</tr>
<tr>
<td>STAR</td>
<td>STAR</td>
<td>$1$</td>
</tr>
</tbody>
</table>
Examples

\{(\text{format,html}), (\text{lang,en})\} \quad \{(\text{format,html}), (\text{lang,en}), (\text{loc,uk})\} \quad \{(\text{img,wbmp}), (\text{lang,en}), (\text{loc,uk})\}
Problems and Solutions

- Selection of undesired variants
  - value prefixes (+, –) denote required and illegal matches
  - examples: <img, +wbmp>, <user, –fred>

- Tie-breakers for ambiguous matches
  - weight factors $w(n)$ for context dimensions
  - weight factors for matching classes (atom, set, range, wildcard)
  - handling of under and over-specified variants

- General scoring function

\[
 f_s(C_1, C_2) = \frac{1}{|N|} \sum_{n \in N} (w(n) \times f_i(n, C_1, C_2)) \times \prod_{n \in N} f_{\pm}(n, C_1, C_2)
\]

\[
 f_{\pm}(n, C_1, C_2) = \begin{cases} 
 1 & \exists c_1 \in C_1, c_2 \in C_2 : \text{name}_1 = \text{name}_2 = n \land \text{value}_1 \cong_{\pm} \text{value}_2 \\
 0 & \text{otherwise.}
\end{cases}
\]
Query Processing Modes

- **Local**
  - match context for every object individually
  - risk of "inconsistent" result sets
  - easy to integrate into existing query processor

- **Local with "convergence"**
  - match context for every object individually
  - add "unmatched" variant context to context state
  - result depends on the structure of the query tree

- **Global**
  - match context for all objects of the query tree
  - optimal, consistent and stable results
  - computationally complex
Limitations and Issues

- Sorted lists to specify user preferences
  - `<lang=de_ch, de, en, it, fr>`
  - update indicator function, makes matching more complex
- Logical expressions to specify complex conditions
  - `(lang=it && loc=ch) || (lang=it && loc=it)`
  - rethink notion and representation of context!
- Scalability
  - applications with large context space
  - information retrieval solutions (vector model, similarity measures)
- However, experiences so far do not suggest major problems related to these issues and limitations
Implementation
Outlook and Future Work

- Integration of context into query language
- Indexing of context-aware data
- Context-aware metadata
  - OMS represents everything as objects
  - collections, associations and methods already context aware
  - investigation of types, type hierarchies, collection hierarchies
- Context-aware programming
  - virtual method dispatching based on signature and context
  - prototype implementation in Prolog
Literature


Next Week
Course Review

• Summary and Exam Information
• Ongoing Research Projects
• Student Projects