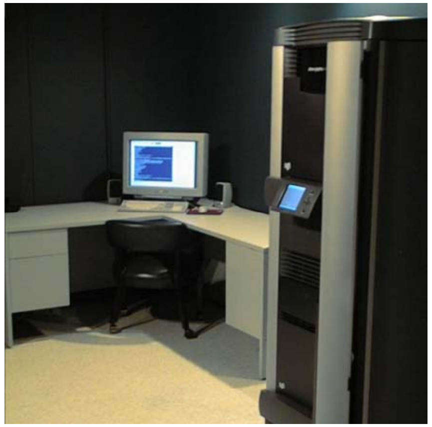
Object-Oriented Databases
Support for Context-Aware Data Management

- 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

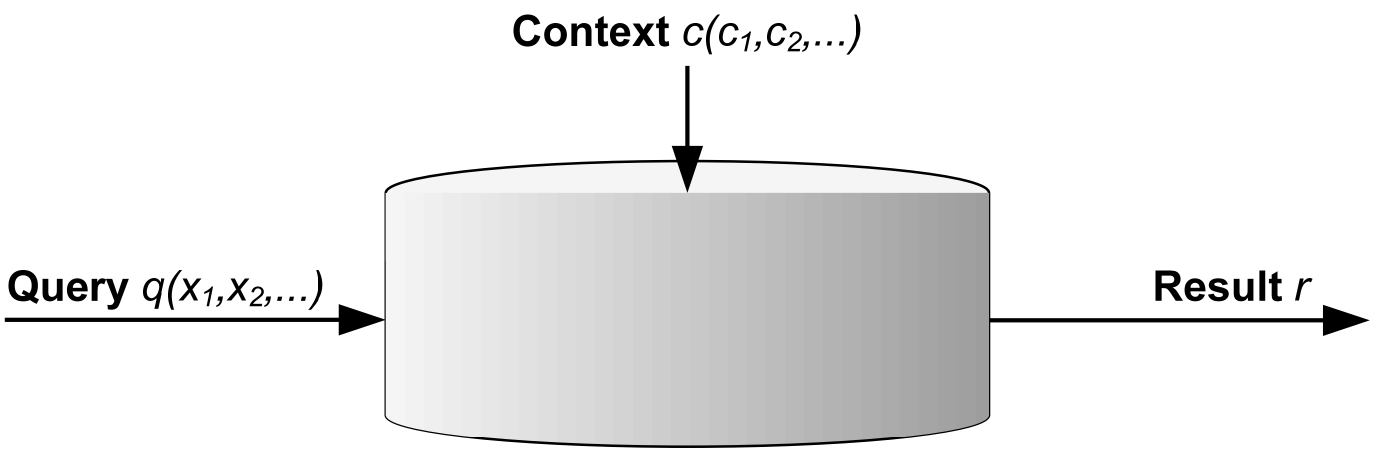
Positioning of Our Work

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

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
  - $c = \langle name, value \rangle$, where $name \in$ NAMES and $value \in$ VALUES
- Context
  - $C(S) = \{\langle name_1, value_1 \rangle, \langle name_2, value_2 \rangle, ..., \langle name_m, value_m \rangle\}$
    $= \{c_1, c_2, ..., c_m\}$
    $\forall\ i: 1 \leq i \leq m \Rightarrow name_i \in S$ and $\forall\ c_i, c_j \in C: c_i = c_j \Rightarrow name_i = name_j$
- Context state
  - special context $C_\star(S)$, $\forall\ name \in S: \exists\ \langle name, value \rangle \in C_\star(S)$

Revisions and Variants

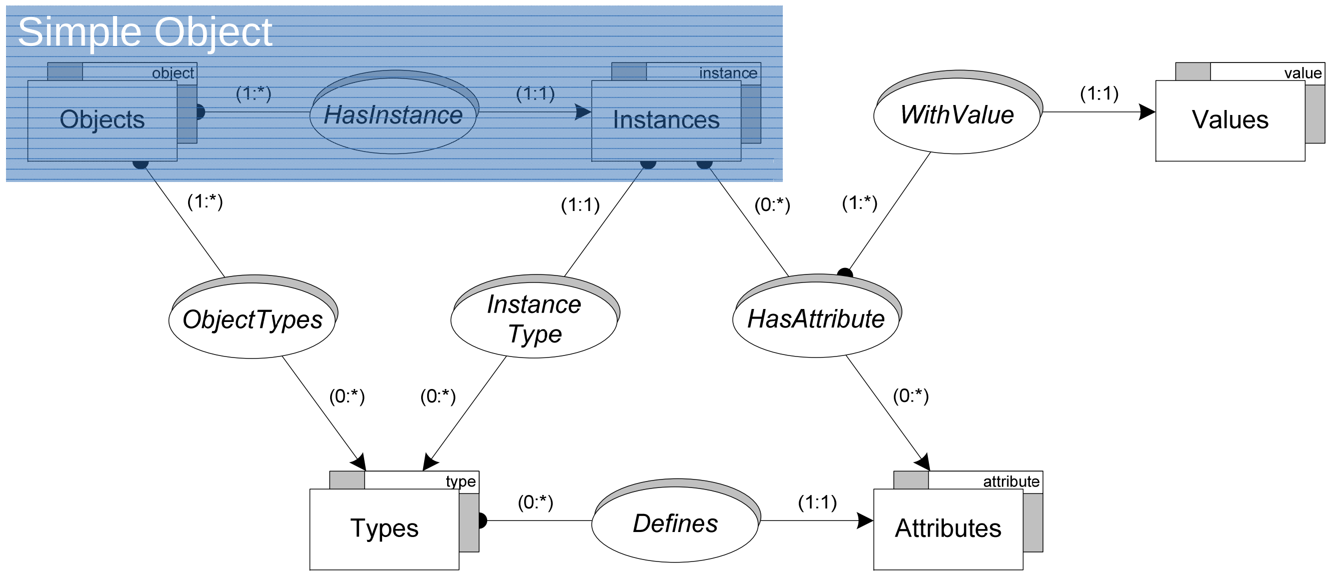
Revisions

- 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

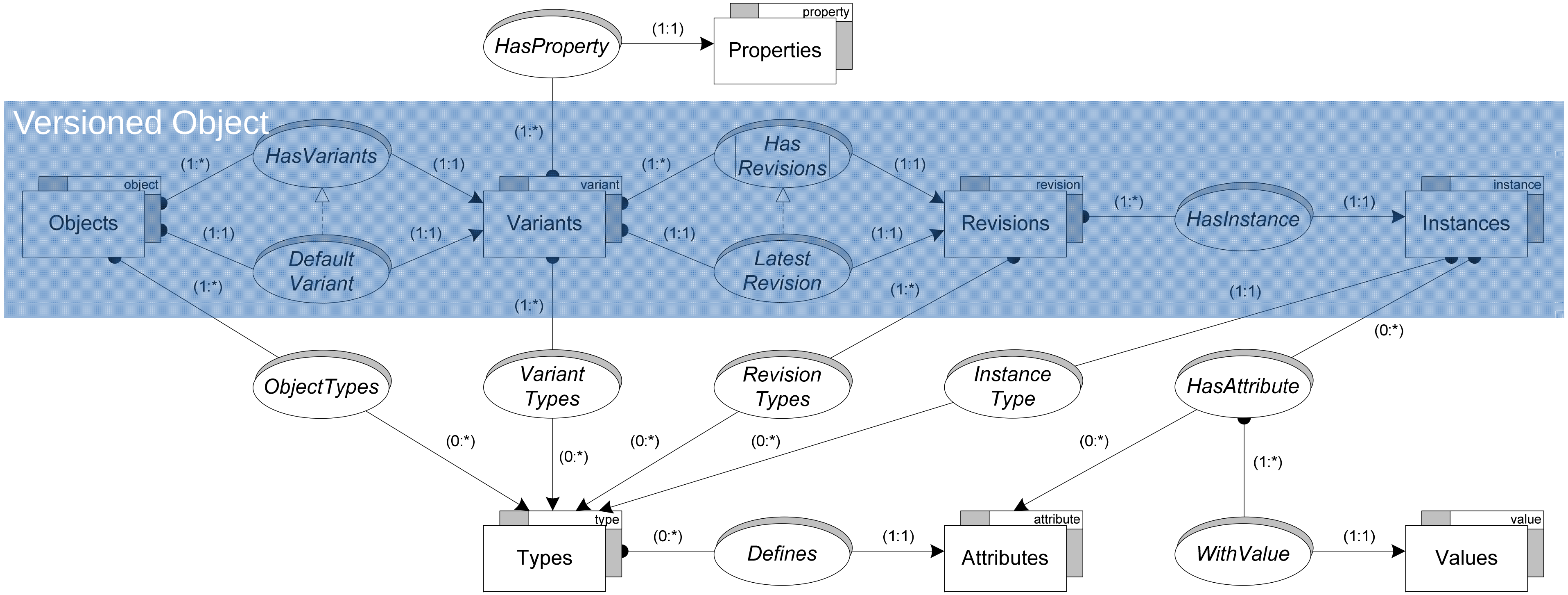
OM Object Metamodel



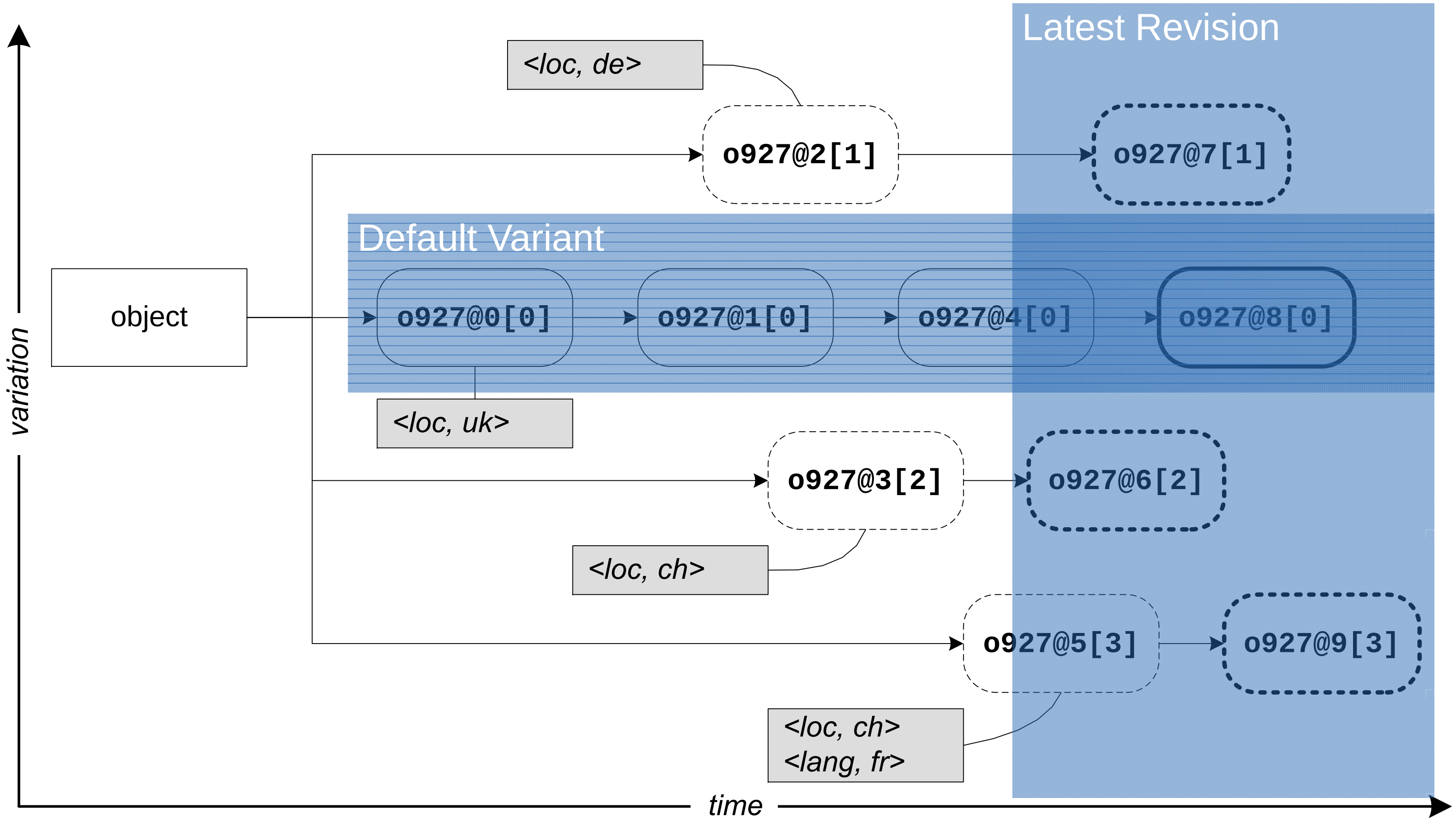
Version Model

- 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: **o** *"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
  - **o927[2]** ► **o927@4[2]** *default version*
  - **o927@3** ► **o927@3[0]** *default variant*
  - **o927** ► **o927@4[0]** *default version and variant*

Context-Aware Query Processing

$\textsc{Match}(o, C_\star(S))$

$V_0 \leftarrow rng(\mathsf{HasVariants}\ dr(\{o\}))$

$V_1 \leftarrow V_0 \propto (x \rightarrow (x \times rng(\mathsf{HasProperty}\ dr(\{x\}))))$

$V_2 \leftarrow V_1 \propto (x \rightarrow (dom(x) \times f_s(C_\star(S), rng(x))))$

$s_{max} \leftarrow max(rng(V_2))$

$V_3 \leftarrow V_2 \,\%\, (x \rightarrow rng(x) = s_{max})$

if $|V_3| = 1 \wedge s_{max} \geq s_{min}$

  then $v \leftarrow V_3\ nth\ 1$

  else $v \leftarrow rng(\mathsf{DefaultVariant}\ dr(\{o\}))\ nth\ 1$

return v

Simple Scoring Function

$$f_{s'}(C_1, C_2) = \frac{1}{|N_1|} \sum_{n \in N_1} f_i(n, C_1, C_2)$$

$$f_i(n, C_1, C_2) = \begin{cases} 1 & \exists\, c_1 \in C_1, c_2 \in C_2 : \\ & \quad name_1 = name_2 = n \wedge value_1 \cong value_2 \\ 0 & \text{otherwise.} \end{cases}$$

- Invoked for every object *o* accessed by the query evaluator
- Scoring function f_s assigns a score value to every variant *v* of *o*
- Indicator function f_i uses matching condition (≅) 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

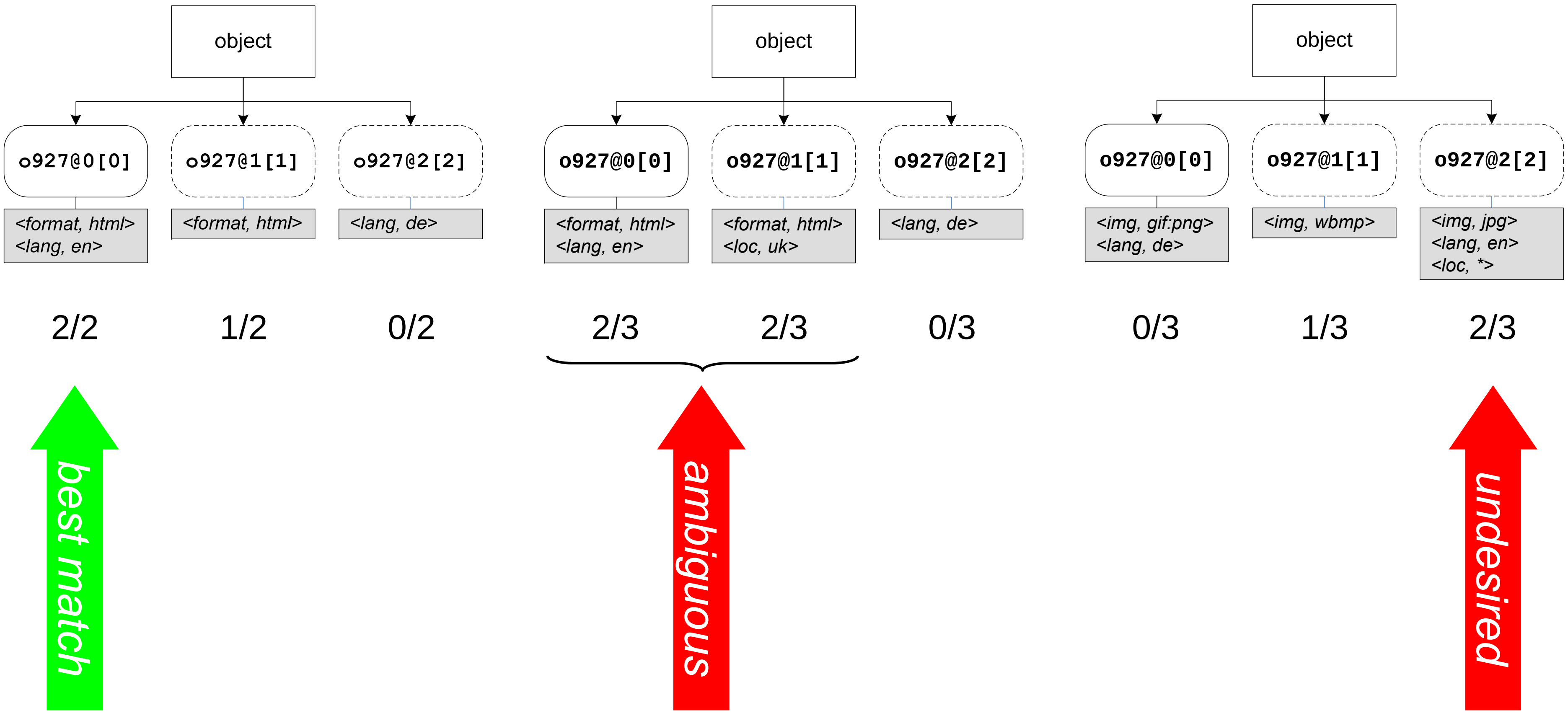
Type	Syntax	Description	Examples
atom	x	Atomic value	**en**, **27**
set	$x_1\{:x_i\}$	Set of atomic values $S := \{x_1, \ldots, x_n\}$	**at:ch:de**, **red:blue**
range	$x_{min}..x_{max}$	Range of atomic values $I := [x_{min}, x_{max}]$	**5.5..7.0**, **a..f**
star	*	Wildcard	*

Matching Condition (≅) for Context Values

x	y	Matching Condition
ATOM	ATOM	$x = y$
ATOM	SET	$x \in y$
ATOM	RANGE	$y_{min} \leq x \leq y_{max}$
ATOM	STAR	$\top$
SET	ATOM	$y \in x$
SET	SET	$x \cap y \neq \emptyset$
SET	RANGE	$\exists\, k \in x : y_{min} \leq k \leq y_{max}$
SET	STAR	$\top$
RANGE	ATOM	$x_{min} \leq y \leq x_{max}$
RANGE	SET	$\exists\, k \in y : x_{min} \leq k \leq x_{max}$
RANGE	RANGE	$max(x_{min}, y_{min}) < min(x_{max}, y_{max})$
RANGE	STAR	$\top$
STAR	ATOM	$\top$
STAR	SET	$\top$
STAR	RANGE	$\top$
STAR	STAR	$\top$

Examples

{(format,html), (lang,en)} {(format,html), (lang,en), (loc,uk)} {(img,wbmp), (lang,en), (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

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

$$f_\pm(n, C_1, C_2) = \begin{cases} 1 & \exists\, c_1 \in C_1, c_2 \in C_2 : \\ & \quad name_1 = name_2 = n \wedge value_1 \cong_\pm value_2 \\ 0 & \text{otherwise.} \end{cases}$$

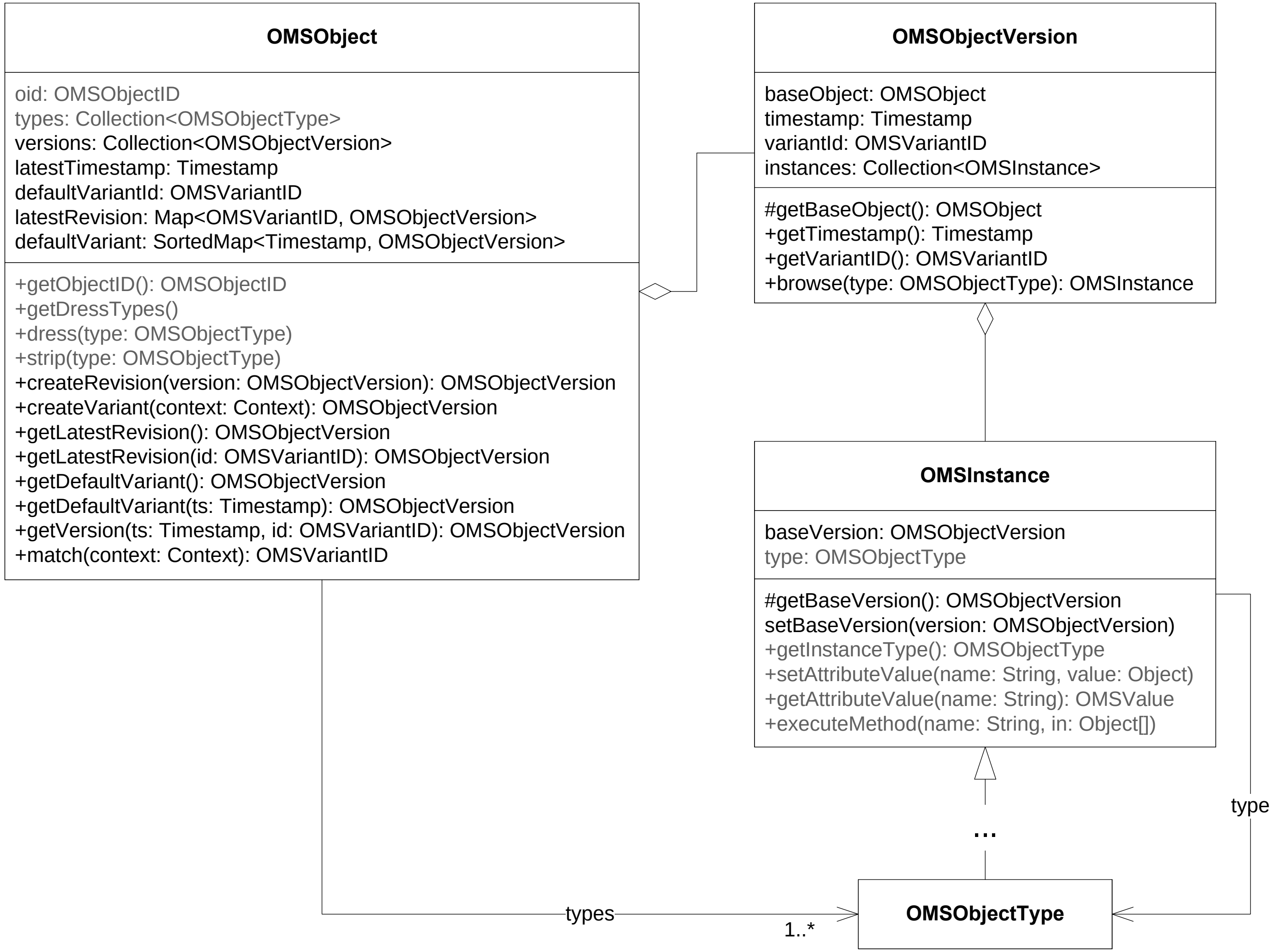
Query Processing Modes

- 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

- *R. Belotti, C. Decurtins, M. Grossniklaus, M. C. Norrie, and A. Palinginis.* **Interplay of Content and Context**, In *Journal on Web Engineering, Vol. 4, No. 1, 2005*
- *M. C. Norrie, B. Signer, M. Grossniklaus, R. Belotti, C. Decurtins, and N. Weibel.* **Context-Aware Platform for Mobile Data Management**, In *Wireless Networks, Vol. 13, No. 6, 2007*
- M. Grossniklaus: **An Object-Oriented Version Model for Context-Aware Data Management**, *PhD Thesis, ETH No. 17137, 2007*

Next Week

Course Review

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