ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Context-Aware Data Management

Michael Grossniklaus

Institute for Information Systems

ETH Zürich

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

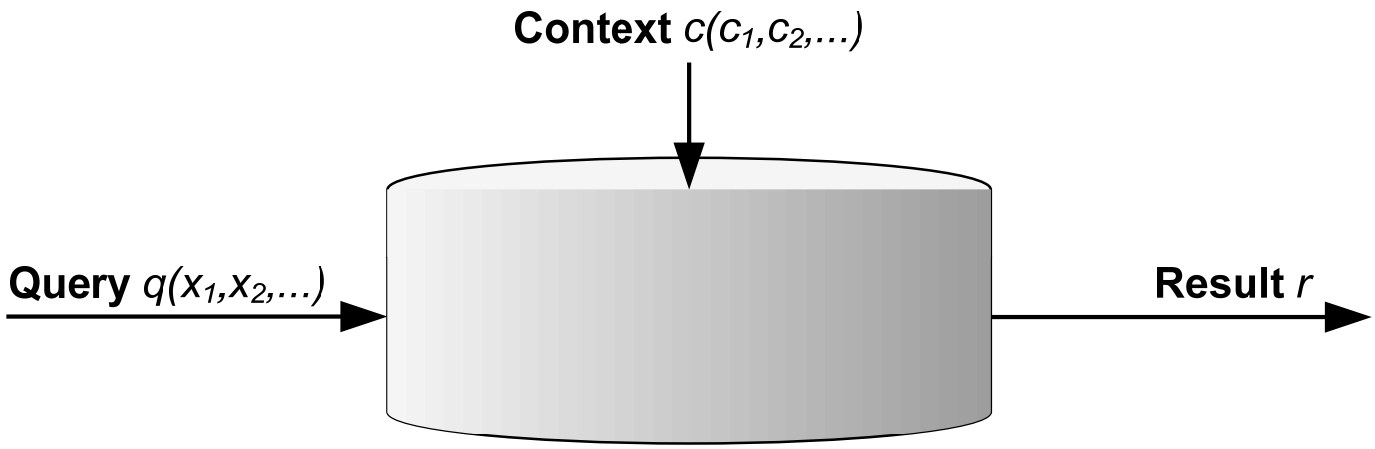
- Models
  - context representation and management
- Infrastructures
  - context gathering, processing, augmentation, management and trigger-based application adaptation
- CASE tools
  - model-based generation of context-aware applications
- Very few context-aware information systems, but...
  - Temporal databases
  - Engineering databases (CAD, CAM)
  - Software configuration management

...have addressed comparable problems in the past

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

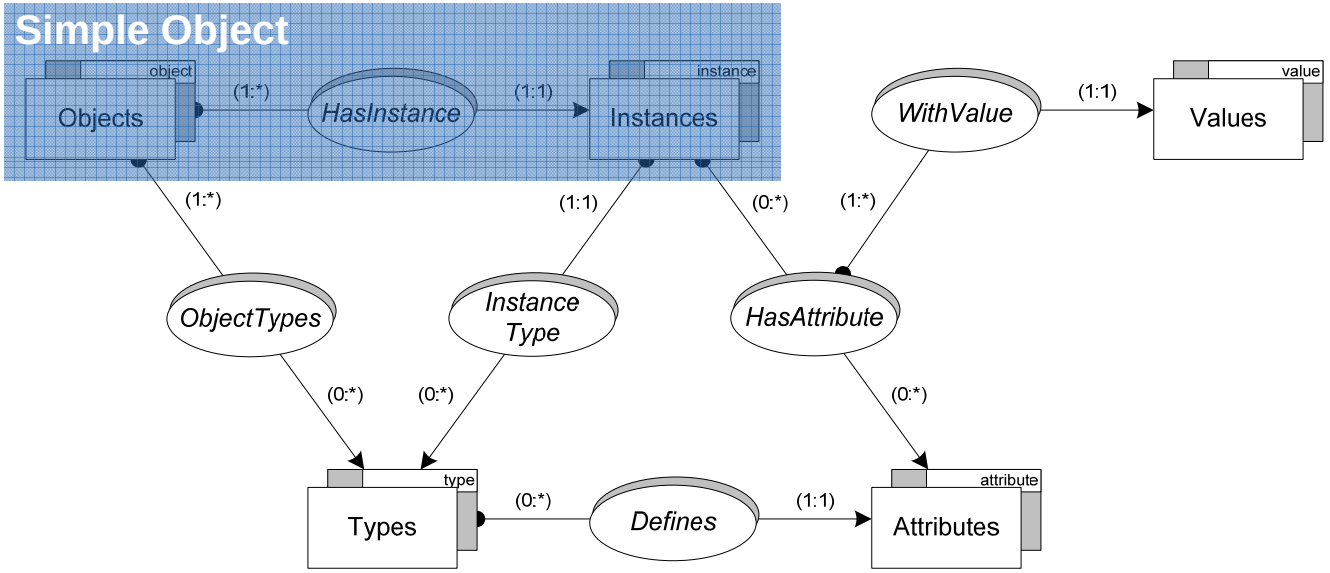
Context, Context Space and Context State

- 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 $C_\star(S)$, $\forall\ name \in S: \exists\ <name, value> \in C_\star(S)$

The OM Data Model

- Extended Entity-Relationship model for object-oriented data management
  - Data represented as objects, i.e. attributes and methods
  - Separates typing and classification
  - Multiple inheritance, multiple instantiation, multiple classification
  - Binary associations as first-order concept
  - Constraints for integrity, classification and evolution
  - Data definition, manipulation and query language OML
  - Defined through a metamodel expressed in OM
- Several implementations exist
  - Metadata managed as objects
  - OMS Pro, OMS/Java, eOMS and OMS Avon

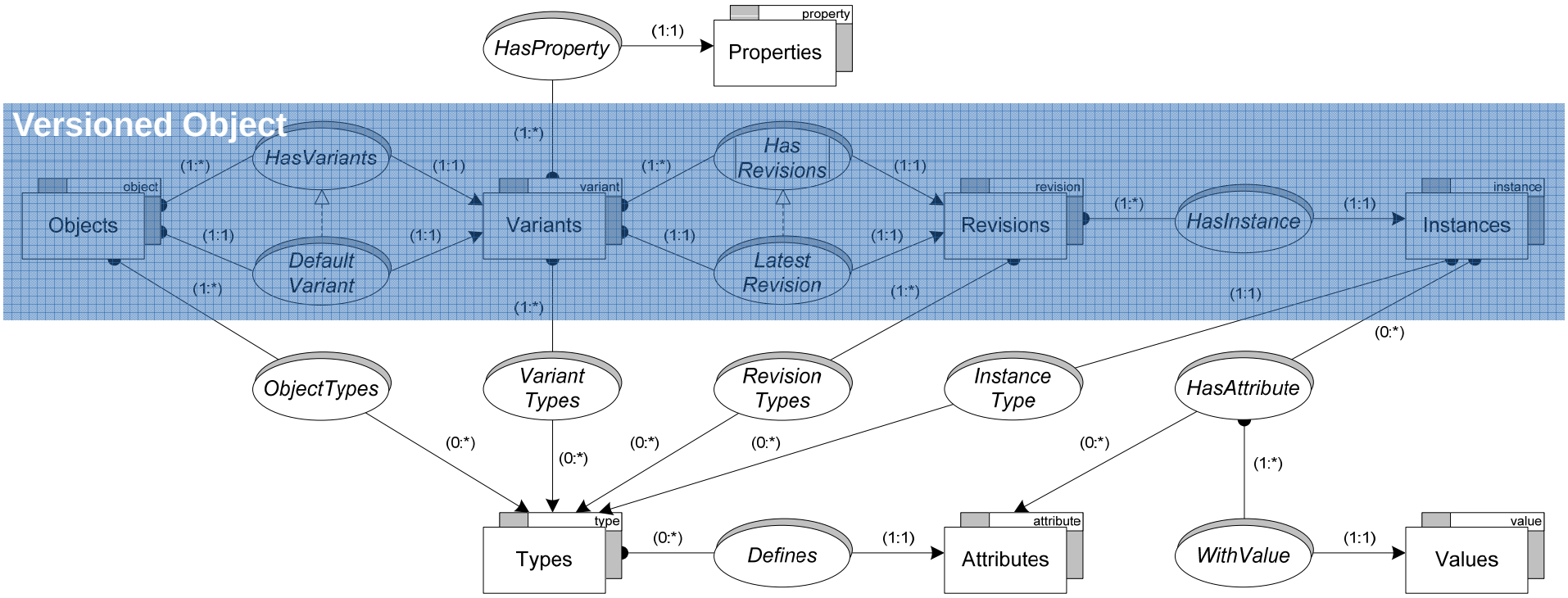
OM Object Metamodel



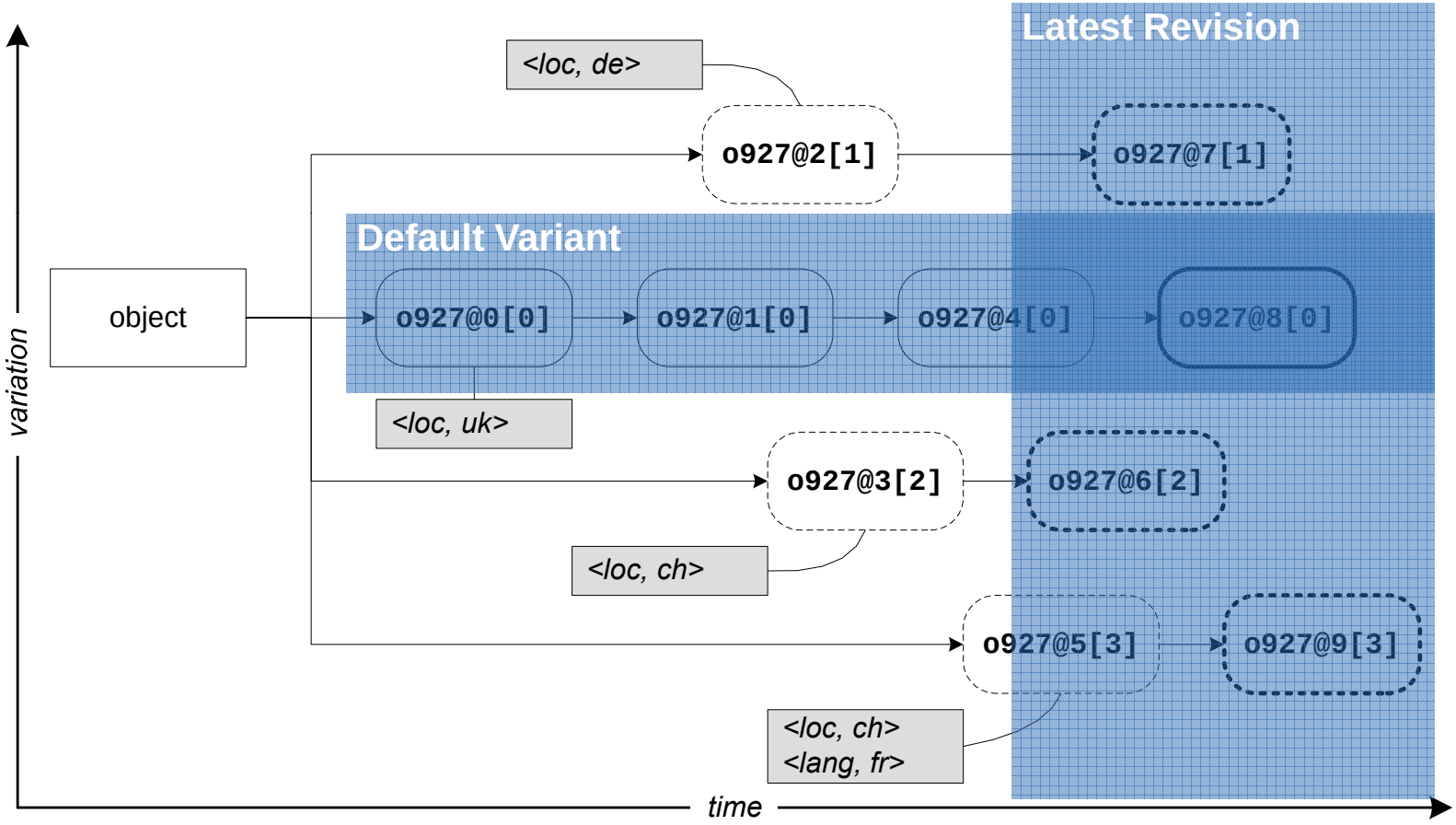
Version Model

- 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

Context-Aware Query Processing

$\text{MATCH}(o, C_\star(S))$
1 $V_0 \leftarrow rng(\textit{HasVariants}\ dr(\{o\}))$
2 $V_1 \leftarrow V_0 \propto (x \rightarrow (x \times rng(\textit{HasProperty}\ dr(\{x\}))))$
3 $V_2 \leftarrow V_1 \propto (x \rightarrow (dom(x) \times f_s(C_\star(S), rng(x))))$
4 $s_{max} \leftarrow max(rng(V_2))$
5 $V_3 \leftarrow V_2\ \%\ (x \rightarrow rng(x) = s_{max})$
6 **if** $|V_3| = 1 \wedge s_{max} \geq s_{min}$
7 **then** $v \leftarrow V_3\ nth\ 1$
8 **else** $v \leftarrow rng(\textit{DefaultVariant}\ dr(\{o\}))\ nth\ 1$
9 **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}

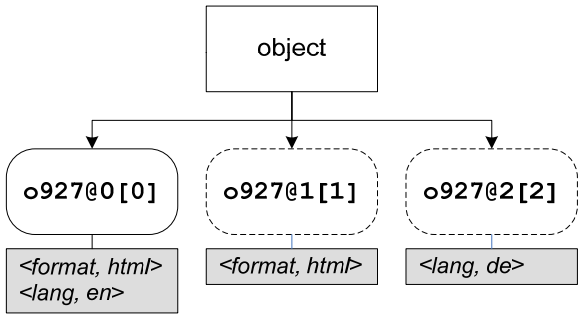
Syntax for Context and Property Values

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 $S := \{x_1, ..., x_n\}$	**`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	**`*`**

Examples

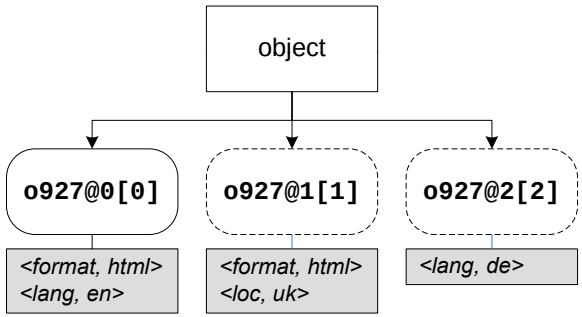
{(format,html), (lang,en)}



2/2 1/2 0/2

best match

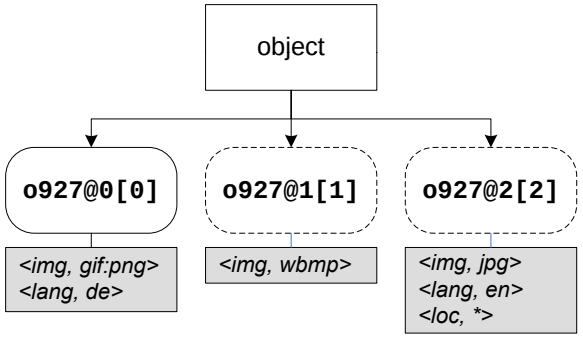
{(format,html), (lang,en), (loc,uk)}



2/3 2/3 0/3

ambiguous

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



0/3 1/3 2/3

undesired

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 for context dimensions
  - Weight factors for matching classes (atom, set, range, wildcard)
  - Handling of under and over-specified variants
- General scoring function

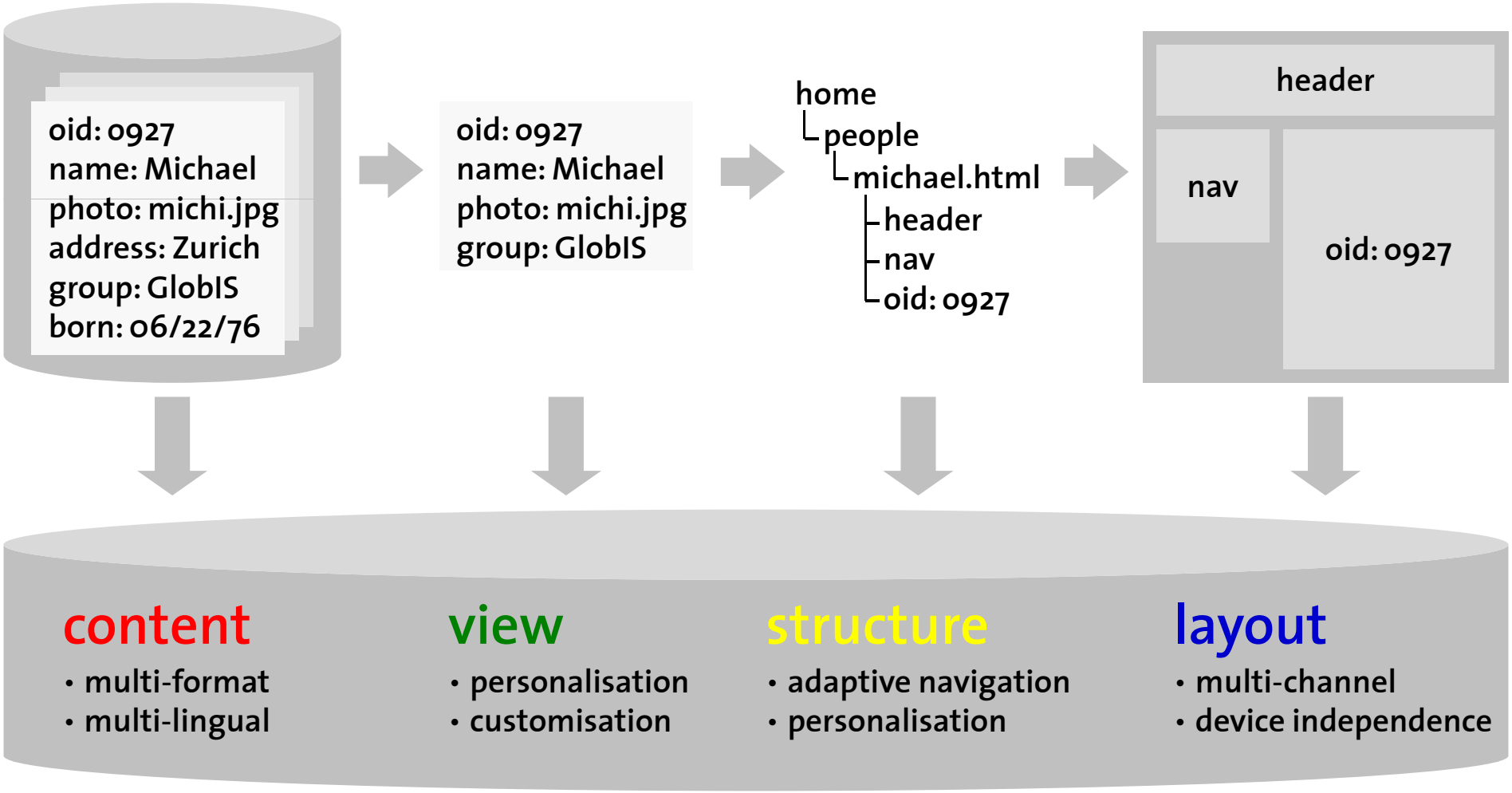
$$f_s(C_1, C_2) = \frac{1}{|N|} \sum_{n \in N} (w(n) \times f_i(n, C_1, C_2)) \times \prod_{n \in N} f_{\pm}(n, C_1, C_2)$$

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

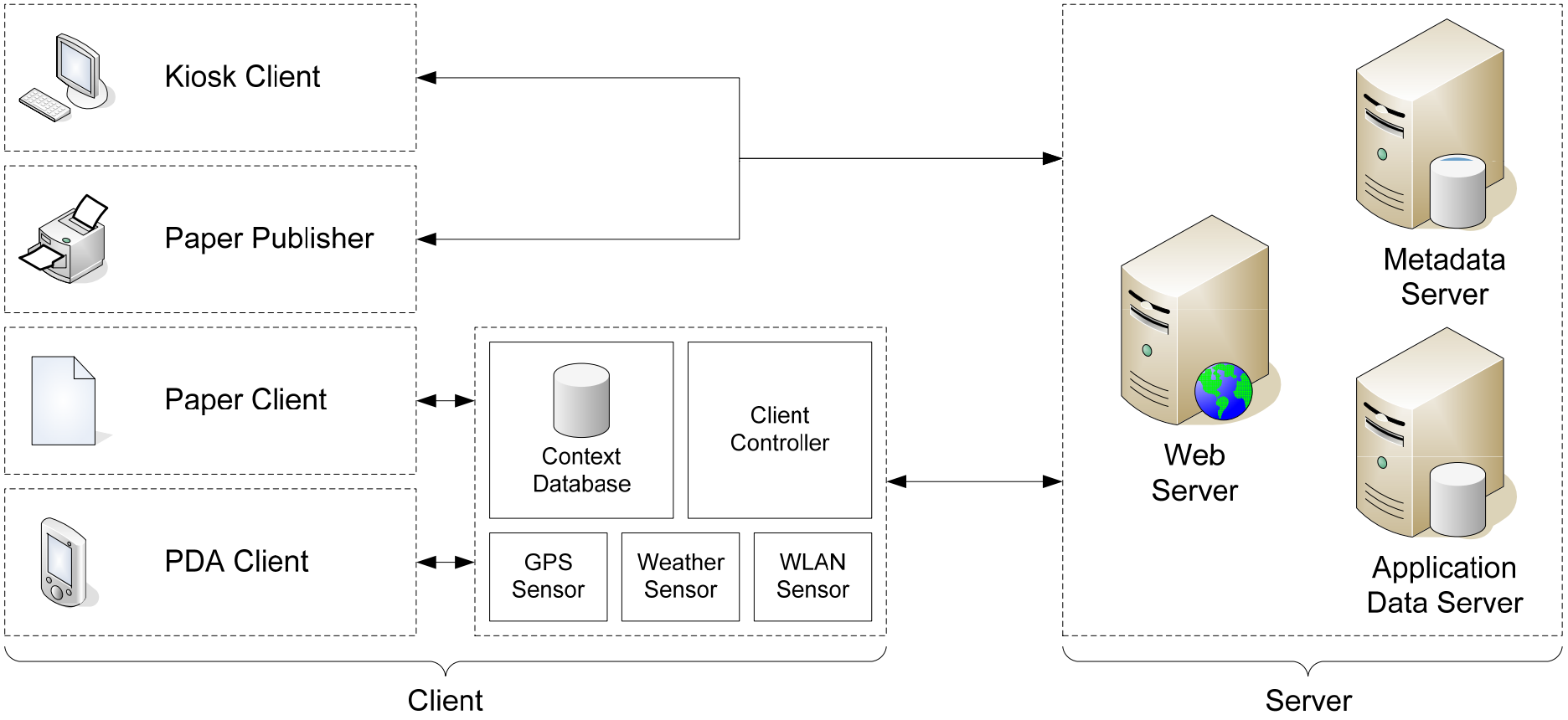
XCM System

- XCM (Extensible Content Management)
  - content application concepts
  - structure composition of content
  - view presentation and aggregation
  - layout rendering
- Uses context-aware data management for its metadata
- Implemented using the Java platform
  - client XCM Sitemanager
  - server XCM System
- Based on OMS/Java and later OMSjp with OMS Pro

XCM Publishing Process

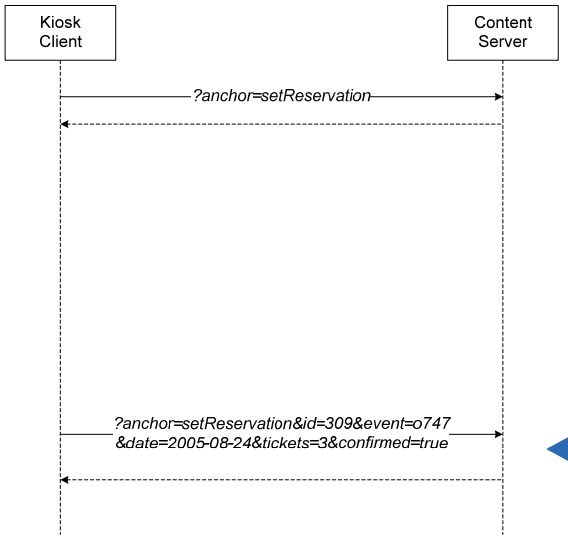


Mobile Tourist Information System



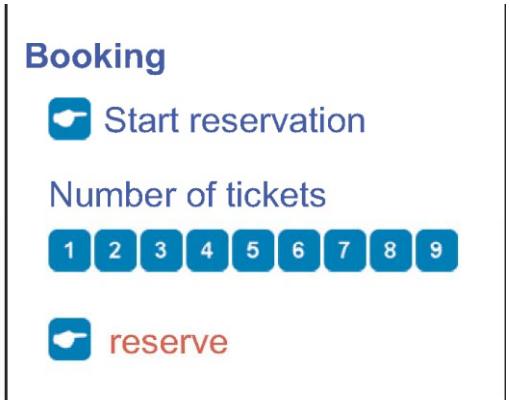
Context-Aware Interaction Processes

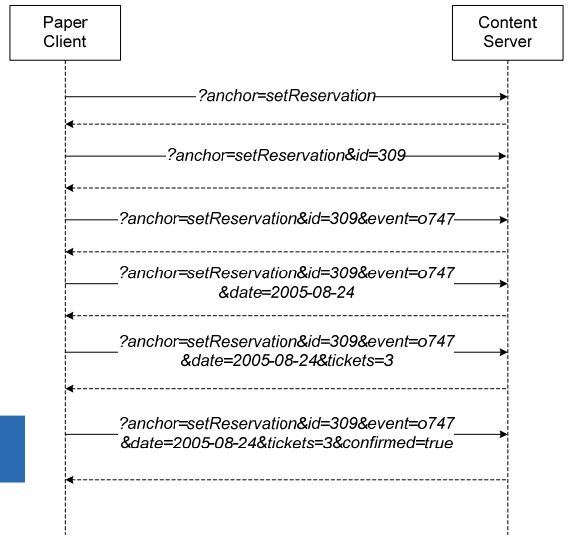
Kiosk Channel



AJAX

Interactive Paper Channel





Context-Aware Methods

Summary and Conclusions

- Context-aware data management
  - Context representation
  - Two-dimensional version model
  - Matching algorithm
- Context-aware content management
  - Separation of content, structure, view and layout
  - Extensibility through object-orientation and inheritance
- EdFest mobile tourist information system
  - Web engineering architecture
  - Multi-modal, multi-channel content delivery

Outlook

- Integration of context into query language
- 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
  - First steps taken in EdFest mobile tourist information system
  - Virtual method dispatching based on signature and context
  - Prolog prototype
- Content management for personal data
  - Apply lessons learnt to improve today's file systems
  - OpenDocument and Office Open XML file formats

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

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