Semantic Web Technologies II

SS 2008

14.05.2008

Semantic Search and Information Integration

Dr. Peter Haase
PD Dr. Pascal Hitzler
Dr. Steffen Lamparter
Denny Vrandecic

Topics

- **Semantic Search**
  - **Overview**
  - Ontology-based Information Retrieval
  - Ontology-based Query Interpretation
  - Natural Language Interfaces
  - Architectural Aspects and Examples

- Information Integration
  - Ontology Mapping
  - Automated Mapping Discovery

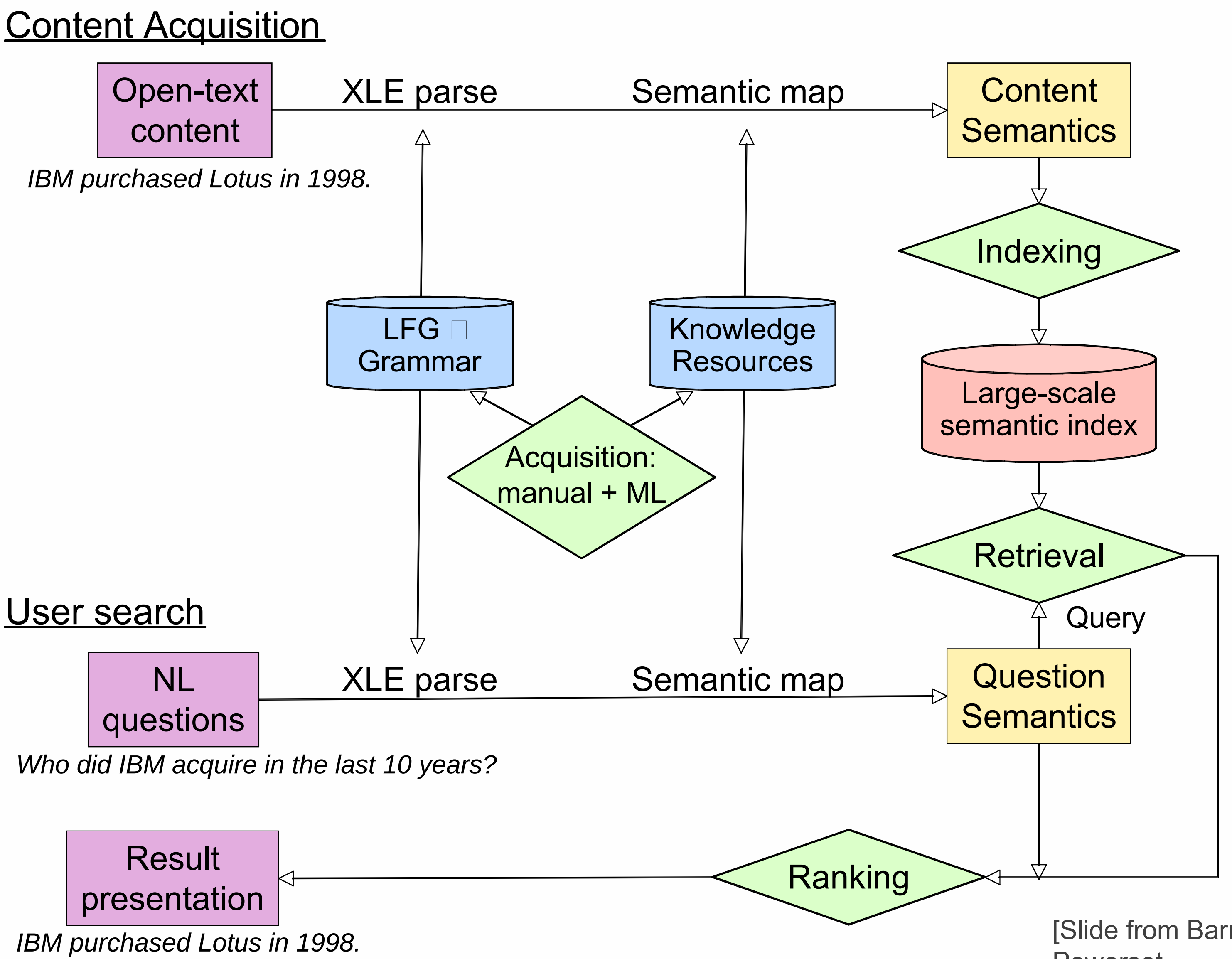
Information Retrieval

Information Retrieval Model is a quadruple $\langle D, Q, F, R(q_i, d_j) \rangle$

- *D* is a set composed of views (representations) for the **resources** (documents) in the collection.
- *Q* is a set composed of views (representations) for the **user information needs** called queries.
- *F* is a framework for modeling **resource representations, queries and their relationships**.
- $R(Q_i,D_j)$ is a **ranking** function which associates a real number with a query Q_i and document representation D_j
- Such ranking defines an ordering among the documents with regard to the query.

- Search based on classical Information Retrieval
  - Resources are text documents
  - User needs (queries) expressed as keyword
  - Simple syntactic matching of keywords against documents

Powerset: Natural Language Search Architecture



[Slide from Barney Pell], Powerset

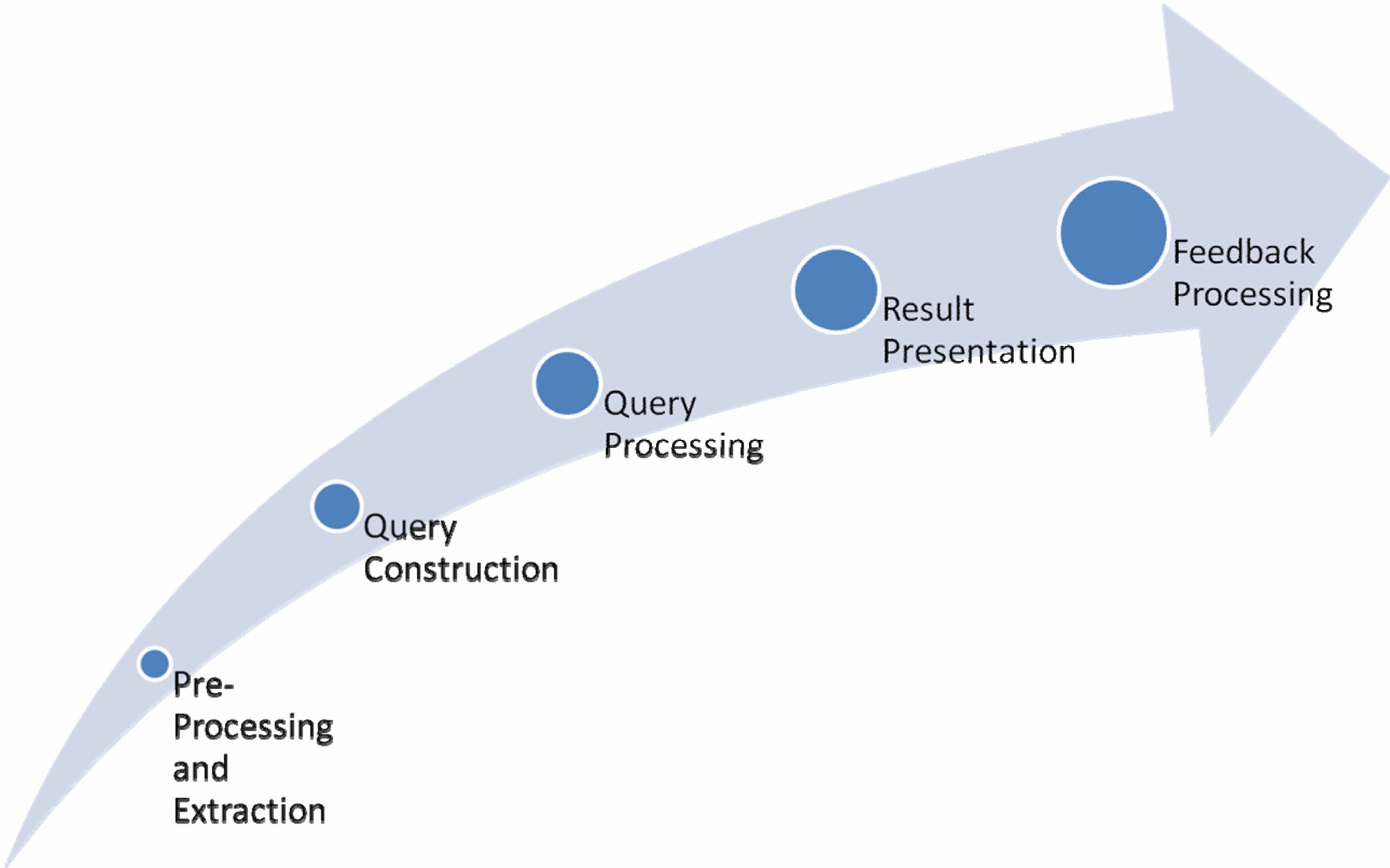
What do we mean by Semantic Search?

- Working definition:

"Semantic Search is a process of information access, where one or several activities can be supported by a set of functionalities enabled by semantic technologies"

- Terminology:
  - Information: documents vs. facts
  - Semantic technologies: knowledge extraction, knowledge representation, reasoning
  - Ontology-enhanced IR vs. ontology-based IR

Semantic Search: a process view



Semantic Search: a process view

Query Construction
Expansion / Refinement

Query Processing: Matching Document with Question
- Type: IR … QA
- Query: Keywords … SPARQL
- Document: Bag of Words … Statements
- Storage: Index … KB
- Approach: Statistical … Ontology-based

Result Presentation
Ranking / Clustering

Feedback Processing
Implicit / Explicit

Pre-Processing
- Tokenizing
- Lemmatizing
- Parsing

Extraction
- Instance/ Concept / Relation
- Expressive Axioms
- Document representation

Semantic Search =
IR + IE + KR + Reasoning ?
What is the right combination?
How to combine?

Query Construction – Task

- **Task:** "Query construction is the first step in the information access process that concerns with assisting the user in the **specification of the information need.**"
- **Result:** "A representation of the information need in terms of primitives of a language supported by the system."
- Language supported by the system
  - keyword-based queries
  - database queries (SQL)
  - XML-based query language (XQuery, XPath, XML Fragments)
  - Semantic Web query languages (SPARQL, DL conjunctive queries, F-logic)
- Enhancement
  - user query is incomplete representation of the user information need
  - modification through expansion, disambiguation and refinement to achieve a more complete and precise representation of the need
- Translation
  - transformation of keywords- or NL-queries to a formal query

Query Construction – Advanced Approaches

- Interpretation of keywords
  - Keywords map to index containing of ontology elements
  - Use **graph exploration** to compute possible connections
  - Map connections to elements of formal query

- Interpretation of Natural Language Queries using domain knowledge
  - Deep parsing query to obtain Part of Speeches (PoS)
  - **PoS maps** against element types of the knowledge base
  - Query elements are mapped against knowledge base instances
  - Typically requires rich lexical models

Query Processing – Task

- **Task:** "Query processing is a step in the information access process where the **need** as specified in the user query **is matched against the system resource model** so as to retrieve the (documents containing the) relevant information."
- **Result**
  - "(A list of ranked documents containing) the information that satisfy the user need."
  - Topical information need → documents about some topics (Document Retrieval)
  - Focused information need → document parts, e.g. section, passage (Focussed IR)
  - Exact information need → an answer to a question (Question Answering)
- Matching procedure is dictated by query and document representation
- "Terms-only" matching
  - Keywords-based queries / "Bag of words" resource models
  - Statistical IR approaches such as vector space model, probabilistic model, DFR etc.
- Incorporating syntactic information
  - Syntactic properties of the text language → Language Model
  - Syntactic properties of the query and resource representation → XML retrieval
- Incorporating semantic information
  - Representation of query and resources enhanced with ontology elements → ontology-enhanced IR
  - Representation of query and resources based on ontology elements → ontology-based IR

Query Processing – Approaches

- Ontology-enhanced IR [SIGIR07b]:
  - articulates the types of knowledge important for IR for a domain
  - Sacrifice breadth for depth: instantiate this general framework in the **restricted and well-defined domain** of clinical medicine based on the principles of evidence-based medicine (EBM)
  - retrieval conceived as “semantic unification” between needs **expressed in a PICO frame** and **corresponding structures extracted from MEDLINE abstract**
- Ontology-enhanced XML Retrieval [SIGIR07a]
  - high precision strategy using annotated documents: XML-based representation enhanced with semantic tags n**amed entities** and **relations**
  - and XML Fragment for semantic search to **conceptualize**, **restrict**, **relate** terms in the query and **nesting** of relation and entity annotations
- Logic-based Information Retrieval [ECIR05a]
  - A fully logic-based approach where query represented in terms of propositional clauses (DNF) is matched against resources also represented in terms of DNF
  - Matching: entailment too strict → polynomial time algorithm for **Propositional Logic and Belief Revision** that computes **non-binary measure of entailment** using distance between models

Result Presentation – Task

Ranking

- **Task:** “Finding appropriate **measures of relatedness** and use them to **rank results.**”
- **Result:** “A score for each of the result is calculated and results are sorted accordingly.”
- Kinds of relatedness
  - Lexical nearness of terms →synonyms, hyponyms, etc.
  - Topical nearness → buzzwords (politics, weather, etc)
  - Structural nearness → classical distance measure based on bag of words model
  - Ontological nearness → Concepts, Relations, Attributes, Instances
  - Heuristics → hyperlinks (Google’ page rank); anchor text, meta tag, page title,
- Ontology-Driven Semantic Ranking for natural language disambiguation
  - Using conceptual distance
    - Minimal path between concepts
    - Distance to common super concept

Result Presentation – Task

Clustering

- **Task:** "Group and label a given collection of patterns into meaningful clusters."
- **Result:** "Thematically related documents are grouped together in the same clusters."
- Clustering with background knowledge
  - Background ontology where terms matched to ontology elements
  - Integration of ontological knowledge in vector model, i.e. extending document term vector with additional dimensions representing ontological knowledge

Feedback Processing – Task

- **Task:** “The processing of user feedbacks aims to **exploit information from the interactions** to further satisfy the user information need.”
- Relevance feedback as standard paradigm
  - use information about which results of a query are perceived relevant for
    - – tuning system parameters
    - – suggesting new query (query refinement)
  - Feedback can be explicit or implicit (user behaviour)
  - Example: augment query with terms of relevant documents
  - Example: use feedback for disambiguation of query terms

Pre-Processing and Extraction – Task

- **Task:** "Pre-processing and extraction are **offline** tasks that are required to **develop a representation of the resources** available in the system."
- **Result:** "A model supported by the system that captures the information content contained in the resources."
- Model supported by the system
  - Bag-of-words
  - Structured model representation (XML documents)
  - Ontology-based model representation
- The more sophisticated the system resource model, the harder is pre-processing and extraction
  - Bag-of-words mostly developed using
    - tokenization
    - Lemmatization only.
  - Identification of syntactic and semantic information from the resources
    - PoS parsing
    - Extraction of instances, relations and expressive axioms

Pre-Processing and Extraction – Approaches

- Domain-Specific
  - more sophisticated domains (e.g. Chemistry)
- Semi-Supervised Information Extraction
  - Deriving Taxonomies
  - Relation Extraction from the Web
  - Mining Wikipedia
- „Open“ Information Extraction
  - extract relations that are not user-defined
- Linguistically heavy approaches
  - Deep parsing

Topics

- Semantic Search
  - Overview
  - **Ontology-based Information Retrieval**
  - Ontology-based Query Interpretation
  - Natural Language Interfaces
  - Architectural Aspects and Examples

- Information Integration
  - Ontology Mapping
  - Automated Mapping Discovery

Motivation – Complex Information Needs

- Complex Information Need – a scenario

  "A user is searching the publications of the research institute AIFB using the information portal http://www.aifb.uni-karlsruhe.de *. He might look for a* ***publication*** *that*

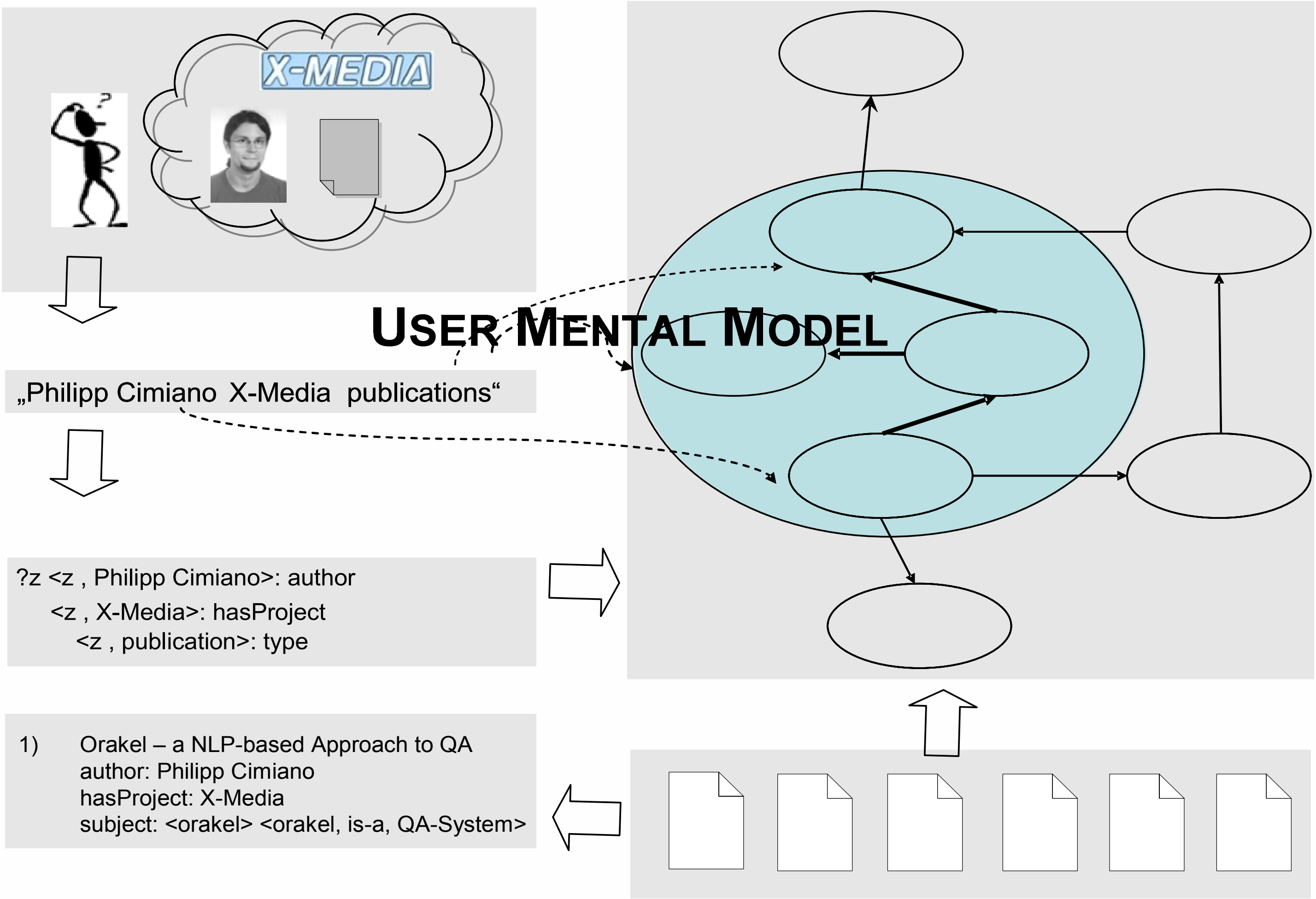
  1. *was written by an* ***author*** *of the knowledge management research group,*
  2. *deals with the* ***topic*** *of information retrieval and*
  3. ***describes*** *a question answering system deployed in a corporate setting."*

- Answering such an information need require more expressiveness
  - More completely interpret the information need
  - **More precisely capture information about resources**

Ontology-based Information Retrieval Model – Components

- Ontology-based Information Retrieval Model
  - Logic-based IR and the use of ontology $\langle D_O, Q_O, F_O, R_O(q_i, d_j) \rangle$
  - Instantiation of the general IR model, i.e. a quadruple
- Resource Model
  - resources are represented through a **set of ontology elements**
- Query Model
  - user information needs are represented as **logical query**
- Matching Framework
  - **Logical entailment**: does representation of the resource entail the ontology-based representation of the information need?
- Ranking
  - Boolean entailment: relevant / not relevant
  - **Non-Boolean entailment** possible

Ontology-based Information Retrieval Model – Process



Query Specification

User Query

Resource Model Ontology

- **Expressive** Resource Description based on conceptual distinction, namely abstract **Content** vs. concrete physical **Content Bearing Object**

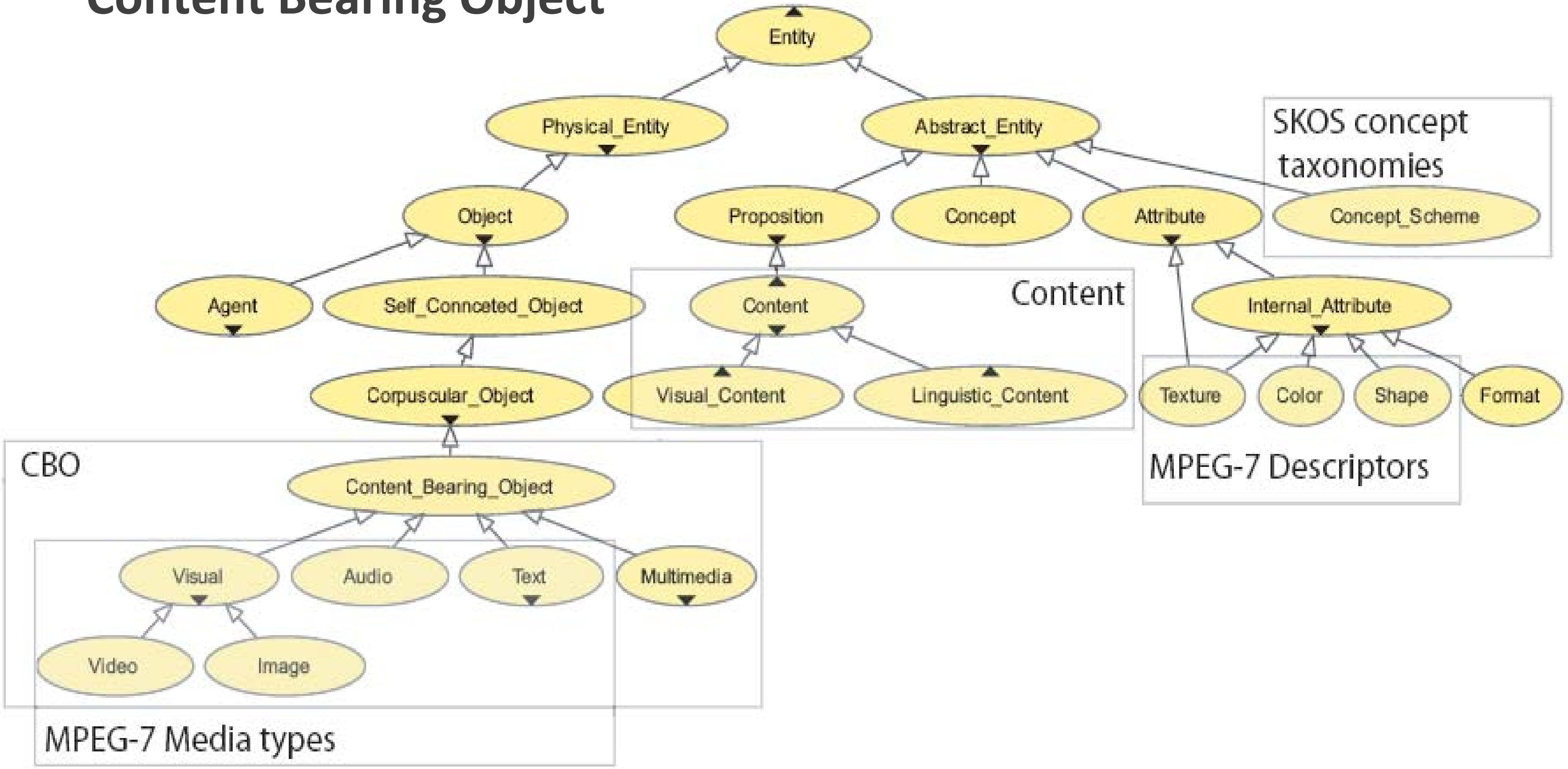


Figure 1: OIRonto Concept Hierarchy

Resource Model – Content Bearing Object

- Content Bearing Object
  - is related to an abstract content entity through contains_information
  - can be materialized in
    - different media types
    - different layouts, color schemes etc.
  - captures
    - **standard metadata**
    - **structure-related** information
    - **presentation-related** information

CBO description

$oir{:}CBO \sqsubseteq$
$\quad \exists oir{:}contains\_information.oir{:}Content \sqcap$
$\quad \exists oir{:}size.xsd{:}byte \sqcap$
$\quad \exists oir{:}format.oir{:}Format \sqcap$
$\quad \forall dc{:}publisher.sumo{:}Agent \sqcap$
$\quad \forall oir{:}creation\_date.xsd{:}date \sqcap$
$\quad \forall dc{:}language.xsd{:}language \sqcap$
$\quad \forall dc{:}title.xsd{:}string \sqcap$
$\quad \forall oir{:}has\_part.oir{:}CBO \sqcap$
$\quad \forall oir{:}is\_part.oir{:}CBO \sqcap$
$\quad \forall oir{:}color.mpeg{:}Color\_Descriptor \sqcap$
$\quad \forall oir{:}shape.mpeg{:}Shape\_Descriptor \sqcap$
$\quad \forall oir{:}texture.mpeg{:}Texture\_Descriptor \sqcap$
$\quad \forall dc{:}rights.sumo{:}Permission \sqcap$
$\quad \forall dc{:}access\_rights.oir{:}Credential$

CBO entity

swrc:InProceedings($pub1492$)
dc:title($pub1492$, "Ontology-based. . . ")
dc:language($pub1492$, "English")
oir:creation_date($pub1492$, "01/02/2007")
oir:author($pub1492c$, $pers98$)

Resource Model – Content

- Content
  - is embodied in some CBO
  - captures
    - **standard metadata,**
    - **structure-related**
    - **content-related information**
    - of the abstract entity
- Expressive description of content
  - content's topic
    - instances of some concept
    - further described by taxonomy
  - content's subject is an entity, which refer to
    - an individual
    - a concept
    - **any complex axioms**

content entity

oir:Content($pub1492c$)
oir:contains_information($pub1492, pub1492c$)

content's topic

oir:topic($pub1492c, top153$)
skos:Concept($top153$)
skos:prefLabel($top153$, "Question Answering")

content's subject

oir:subject($pub1492c, dom{:}id333$)
dom:Corporation($dom{:}id333$)
dom:name($dom{:}id333$, "British Telecom")
oir:subject($pub1492c, dom{:}id555$)
dom:QASystem($dom{:}id555$)
dom:name($dom{:}id555$, "Orakel")

complex subjects' description

dom:OIR $\sqsubseteq$ dom:QA $\sqcap$ dom:IR
dom:deployedAt($dom{:}id555, dom{:}id333$)
oir:is_subject_of(
dom:deployedAt($dom{:}id555, dom{:}id333$),
$pub1492c$)
oir:is_subject_of(
dom:OIR $\sqsubseteq$ dom:QA $\sqcap$ dom:IR,
$pub1492c$)

content description

$$\begin{aligned}&\text{sumo:Content} \sqsubseteq \\ &\quad \exists \text{sumo:embodied\_in.oir:CBO} \sqcap \\ &\quad \exists \text{oir:author.sumo:Cognitive\_Agent} \sqcap \\ &\quad \exists \text{dc:subject.sumo:Entity} \sqcap \\ &\quad \forall \text{oir:topic.skos : Concept} \sqcap \\ &\quad \forall \text{dc:source.sumo:Content} \sqcap \\ &\quad \forall \text{oir:authoring\_date.xsd:date}\end{aligned}$$

Query Answering

Topical query

SELECT $?r$ WHERE {
$?r\ \ oir\!:\!contains\_information\ ?c\ \ .$
$?c\ \ oir\!:\!topic\ \ ?t\ \ .$
$?t\ \ skos\!:\!prefLabel\ \ \text{'}Information\ Retrieval\text{'}$
}

Metadata query

SELECT $?r$ WHERE {
$?r\ \ rdf\!:\!type\ \ sumo\!:\!Entity\ .$
$?r\ \ oir\!:\!contains\_information\ ?c\ \ .$
$?c\ \ oir\!:\!author\ ?p\ \ .$
$?p\ \ swrc\!:\ affiliation\ \ \ ?g\ \ .$
$?g\ \ swrc\!:\!name\ \text{'}Knowledge\ Management\text{'}$
}

Simple content query (with reasoning)

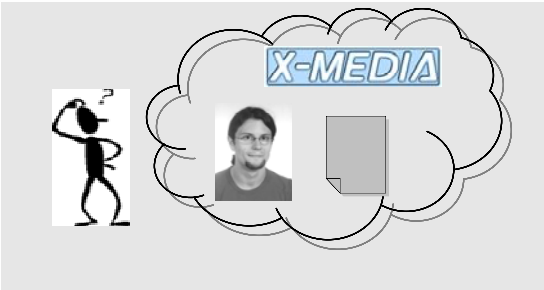
SELECT $?r$ WHERE {
$?r\ \ oir\!:\!contains\_information\ ?c\ \ .$
$?c\ \ oir\!:\!subject\ \ ?s\ \ .$
$?s\ \ rdf\!:\!type\ \ dom\!:\!QASystem$
}

Topics

- Semantic Search
  - Overview
  - Ontology-based Information Retrieval
  - **Ontology-based Query Interpretation**
  - Natural Language Interfaces
  - Architectural Aspects and Examples

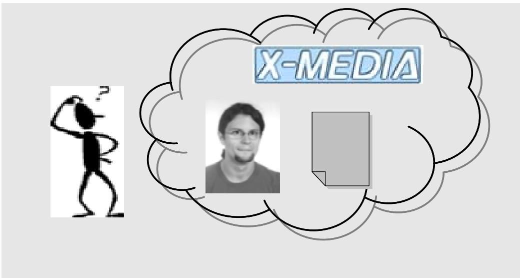
- Information Integration
  - Ontology Mapping
  - Automated Mapping Discovery

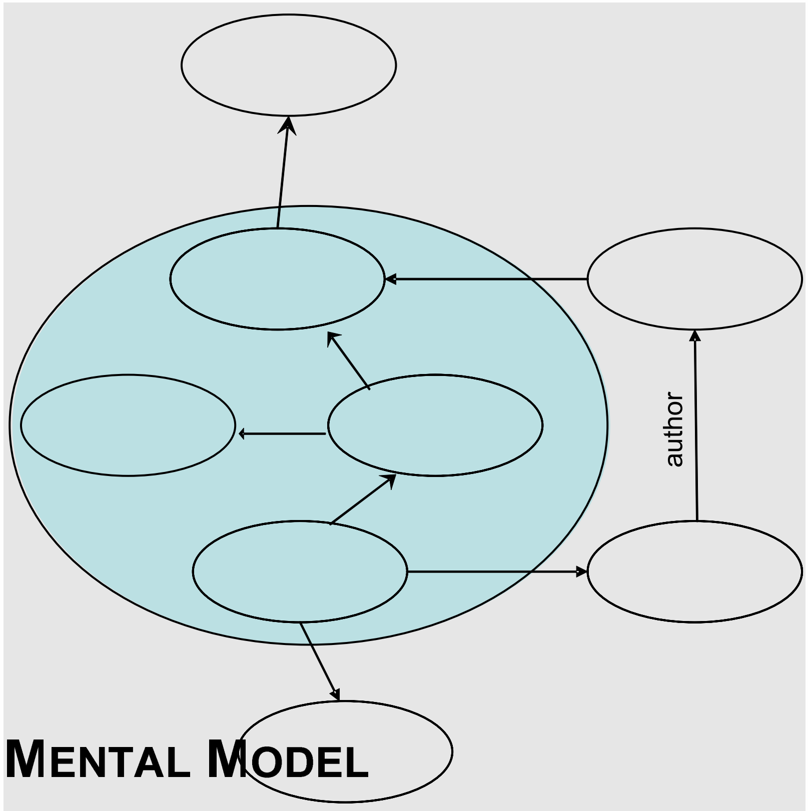
Ontology-based Query Interpretation



USER MENTAL MODEL

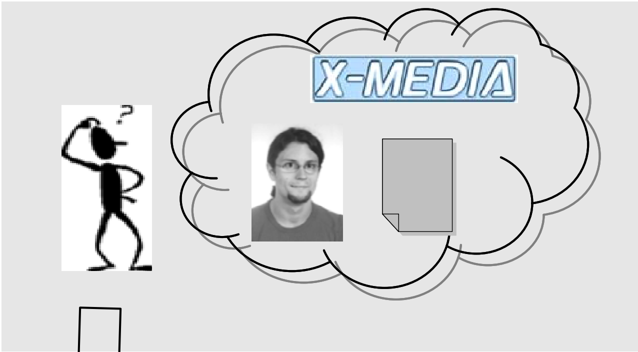
Ontology-based Query Interpretation





USER MENTAL MODEL

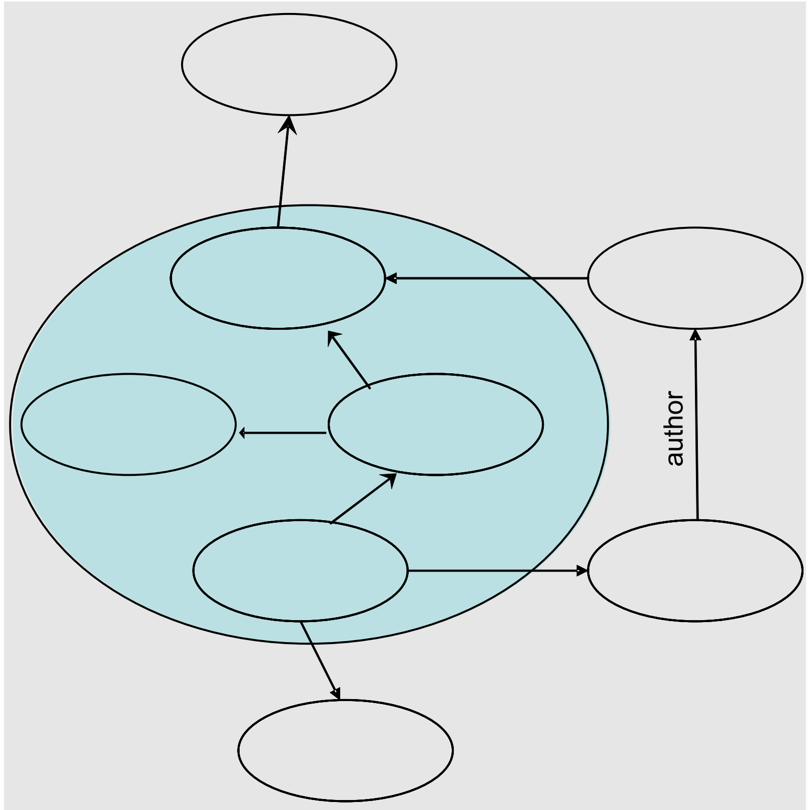
Ontology-based Query Interpretation



USER MENTAL

„Philipp Cimiano X-Media publications“

?z <z , Philipp Cimiano>: author
<z , X-Media>: hasProject
<z , publication>: type

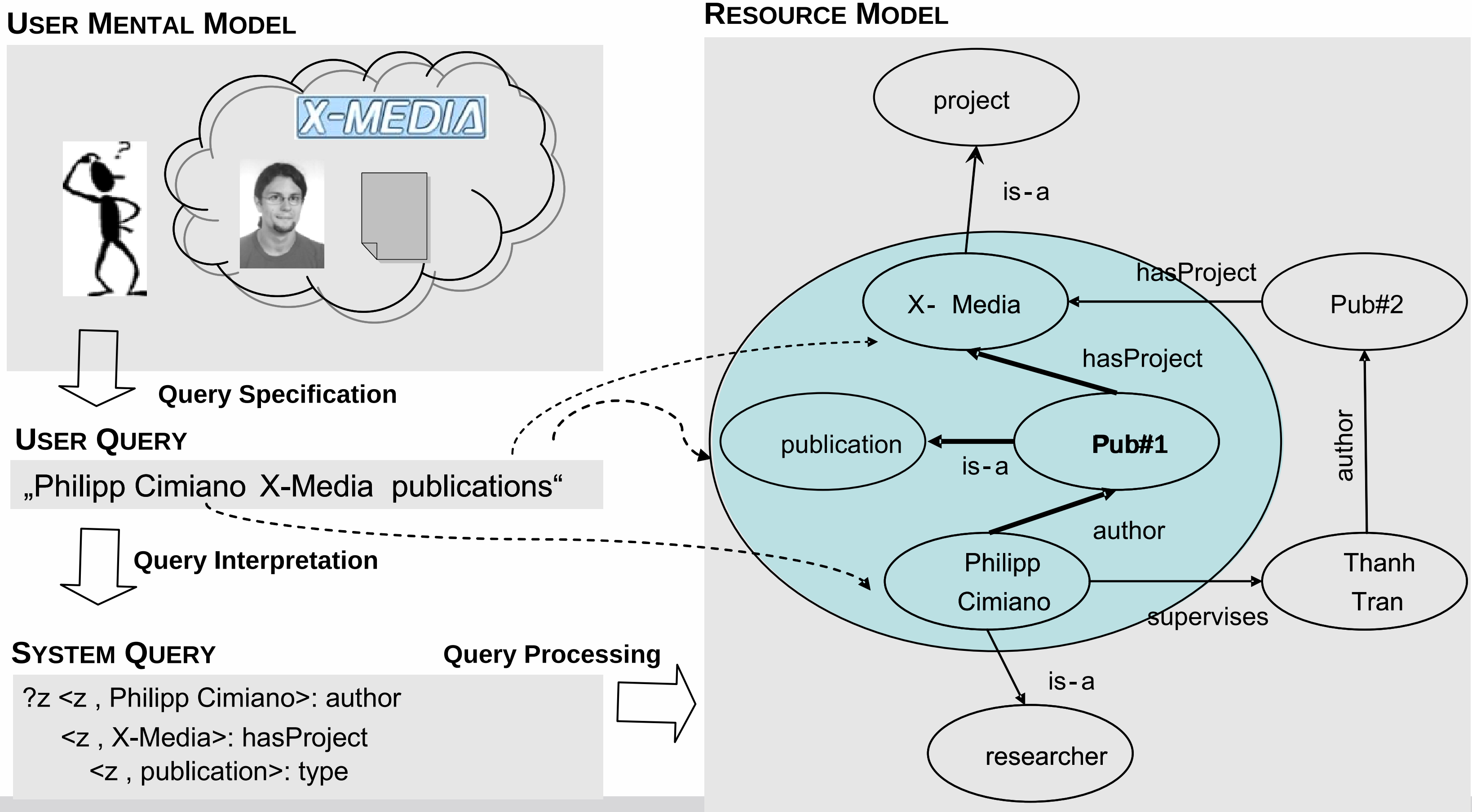


RE

Ontology-based Query Interpretation

Procedure

- Map user query elements to ontology elements
- Explore ontology elements to find connections
- Derive system query from connections



Ontology-based Query Interpretation

Karlsruhe Institute of Technology

AIFB

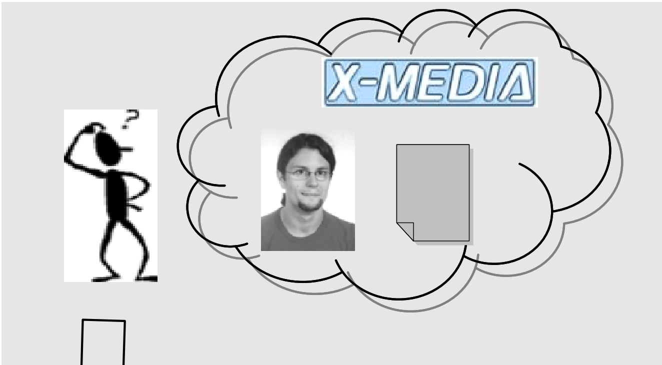
Procedure

- Map user query elements to ontology elements
- Explore ontology elements to find connections
- Derive system query from connections

Assumptions

- Ontology-Mental Correspondence
- Locality of Information Need

USER MENTAL MODEL



Query Specification

USER QUERY

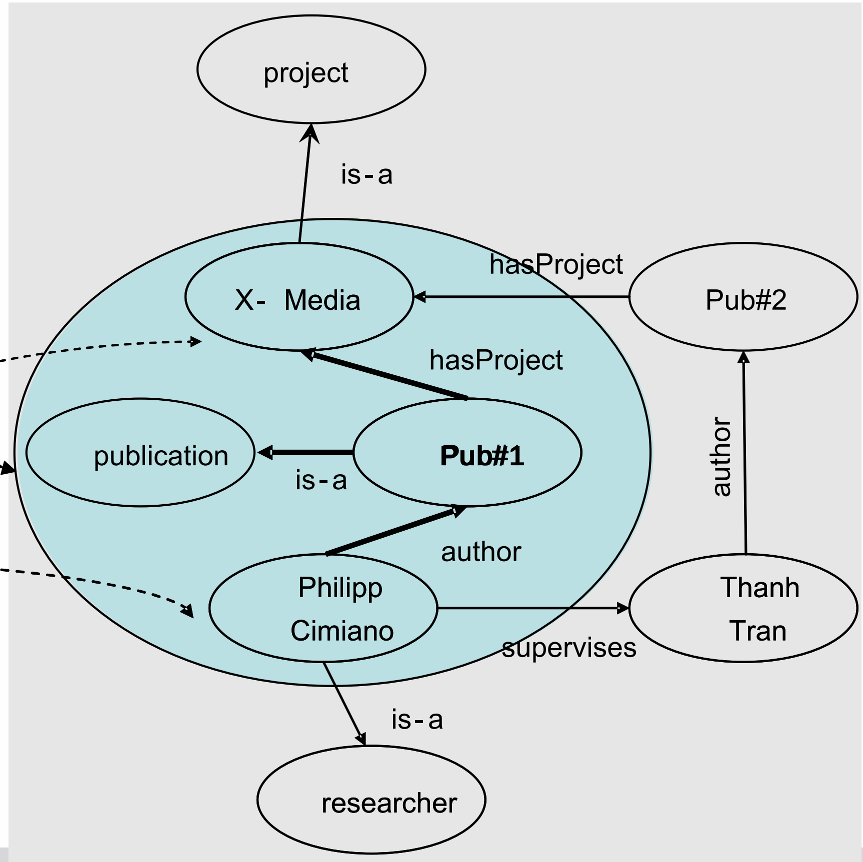
„Philipp Cimiano X-Media publications“

Query Interpretation

SYSTEM QUERY **Query Processing**

?z <z , Philipp Cimiano>: author
<z , X-Media>: hasProject
<z , publication>: type

RESOURCE MODEL

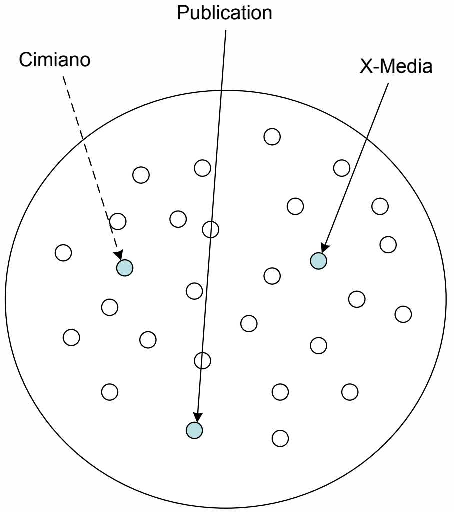


Interpretation of Keywords as Conjunctive Queries

- An instantiation of the generic IR model
- User question **a set of keywords** $Q_U = (k_1, k_2, \ldots, k_n)$
- System query
  - **Conjunction of terms** of the form $x : C$ and $\langle x, y \rangle : R$
  - Where C is a concept, R is a role, and x, y are variables or individuals taken from a set of variable names, or a set of individual names
- System resource model
  - OWL DL knowledge base
  - **Ontology entities:** sets of individuals, data values, concepts, data ranges, object properties and data properties
  - **Connections between entities** are captured by terminological axioms and assertions (concept and property membership)

Step 1 – Mapping Keywords to Ontology Entities

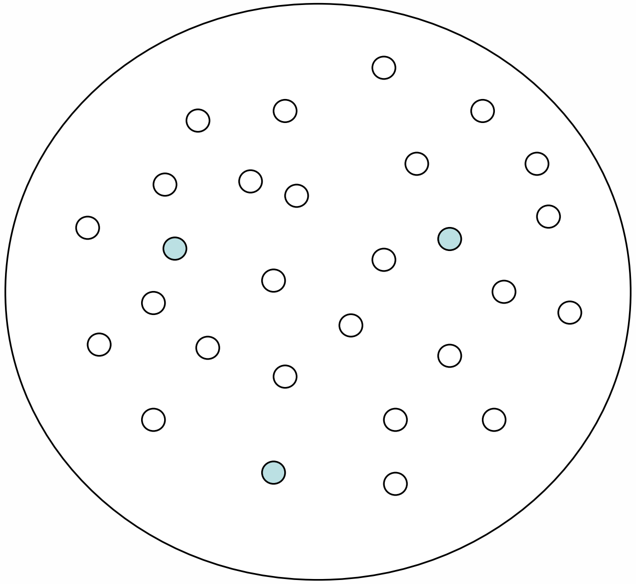
- Match
  - keywords against ontology entities
- "Robust" matching functions
  - **Syntactic variants**
  - **Spelling variants**
- Matching function
  - Index of ontology elements
  - **Fuzzy search** on index with each keyword
  - Return ontology entities ranked according to syntactic similarity



Step 2 – Exploring Connections among Ontology Entities

```
KB Exploration(O′_S, d)
INPUT entities O′_S and traversal width d
OUTPUT graph containing all or some of O′_S
Intitialize new empty graph g
for e ∈ O′_S
do if e is a concept
      then for  all i being instances of e
            do I-P-I Traversal(e, d, g)
      else  if e is an object property
               then for  all i, j with ⟨i, e, j⟩ ∈ O_S
                        do I-P-I Traversal(i, d, g)
                           I-P-I Traversal(j, d, g)
      else  if e is a data property
               then for  all i, j with ⟨i, e, j⟩ ∈ O_S
                        do J-P-I Traversal(j, d, g)
      else  if e is an individual
               then I-P-I Traversal(e, d, g)
      else  if e is a data value
               then J-P-I Traversal(e, d, g)
return g
```

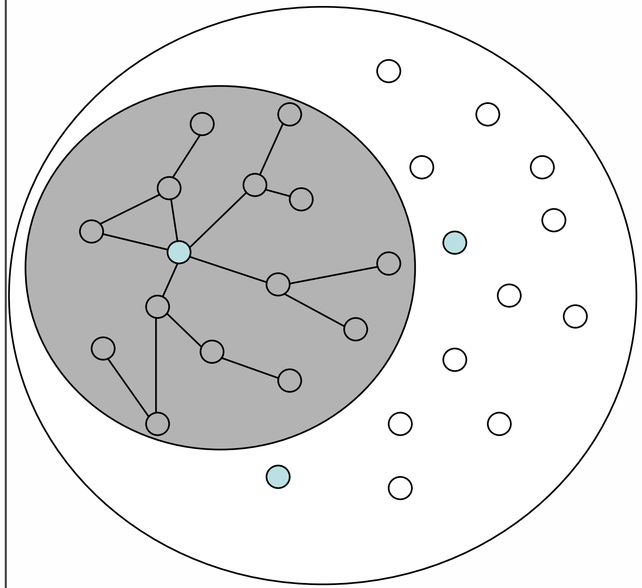
- Algorithms for
  - **Knowledge Base Exploration**
  - Recursive traversal of elements
  - Adopted Depth First Search (DFS) for calculation of paths



Step 2 – Exploring Connections among Ontology Entities

```
I-P-I TRAVERSAL(i, d, g)
INPUT individual i, width d, intermediate graph g
OUTPUT updated graph g
if  i not marked as visited and d > 0
  then
        mark i as visited within O_S
        C_i := {c | i instance of c}
        add edge (i, type, c) to g for all c ∈ C_i
        P := {(i, p, j) | ⟨i, p, j⟩ ∈ O_S}
        for all (i, p, j) ∈ P
        do if  j not marked as visited in O_S
              then add a new edge (i, p, j) to g
                    if  j is an individual
                      then I-P-I TRAVERSAL(j, d − 1, g)
                      else  J-P-I TRAVERSAL(j, d − 1, g)
```

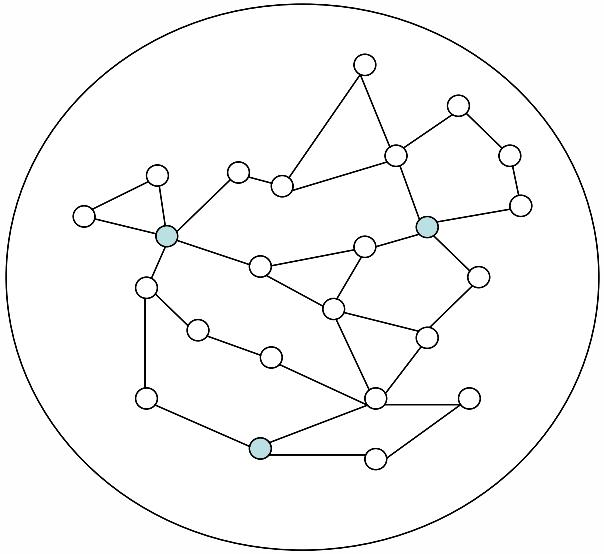
- Algorithms for
  - KB Exploration
  - **Recursive traversal of elements**
  - Adopted Depth First Search (DFS) for calculation of paths



Step 2 – Exploring Connections among Ontology Entities

```
PATH DFS(v, G, E, P, S)
INPUT vertex v, graph G, vertices E, path P, stack S
OUTPUT the path S and labelled edges
Push v into stack S
if v matches any e ∈ E
  then
        if S not already in P
           then
                 Add S to P
                 Empty S and push v into it
        for in- and outgoing edges: e_out(v, w), e_in(w, v)
do if w not already visited
      then if w not already visited
               then Set label of e as "discovered"
                    Push e into stackS
                    PATH DFS(w, G, E, P, S)
                    Pop e from S
               else Set label of e as "back"
Pop v from S
```

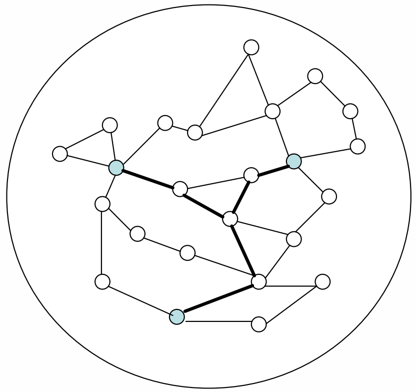
- Algorithms for
  - KB Exploration
  - Recursive traversal of elements
  - **Adopted Depth First Search for calculation of paths**



Step 3 – Deriving Conjunctive Queries from Connections

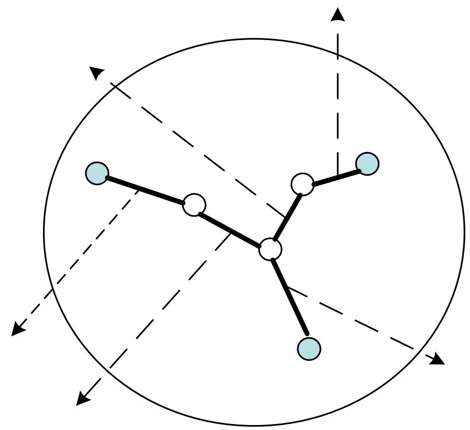
```
CalculateSubGraphs(P, C, R, G, g)
INPUT the paths P by DFS for matching vertices O′s
OUTPUT all subgraphs connecting vertices in O′s
if R = ∅
   then G = G ∪ g
if g = ∅
   then G = newGraph
        for {i, j} ⊆ R
        do for each path p between i and j (as by DFS)
           do add (i,p,j) to G
              CalculateSubGraphs(P\p, C ∪ {i, j}, R\{i, j}, G)
   else for i ∈ R
        do for j ∈ C
           do for for each path p between i and j
              do
                 add (i,p,j) to G
                 CalculateSubGraphs(P\p, C ∪ {i}, R\{i}, G)
```

- **Compute possible subgraphs**
- Mapping Connections to Queries
- Rank Queries



Step 3 – Deriving Conjunctive Queries from Connections

- Compute possible subgraphs
- **Mapping Connections to Queries**
  - concept member connections and property member connections map to corresponding expressions
  - Vertices matching query elements become constants otherwise variables
- Ranking Queries
  - the smaller the length of the path, the more likely is the corresponding interpretation (locality assumption)
  - length of the longest path of the connection graph



Topics

- Semantic Search
  - Overview
  - Ontology-based Information Retrieval
  - Ontology-based Query Interpretation
  - **Natural Language Interfaces**
  - Architectural Aspects and Examples
- Information Integration
  - Ontology Mapping
  - Automated Mapping Discovery

Natural language Interfaces

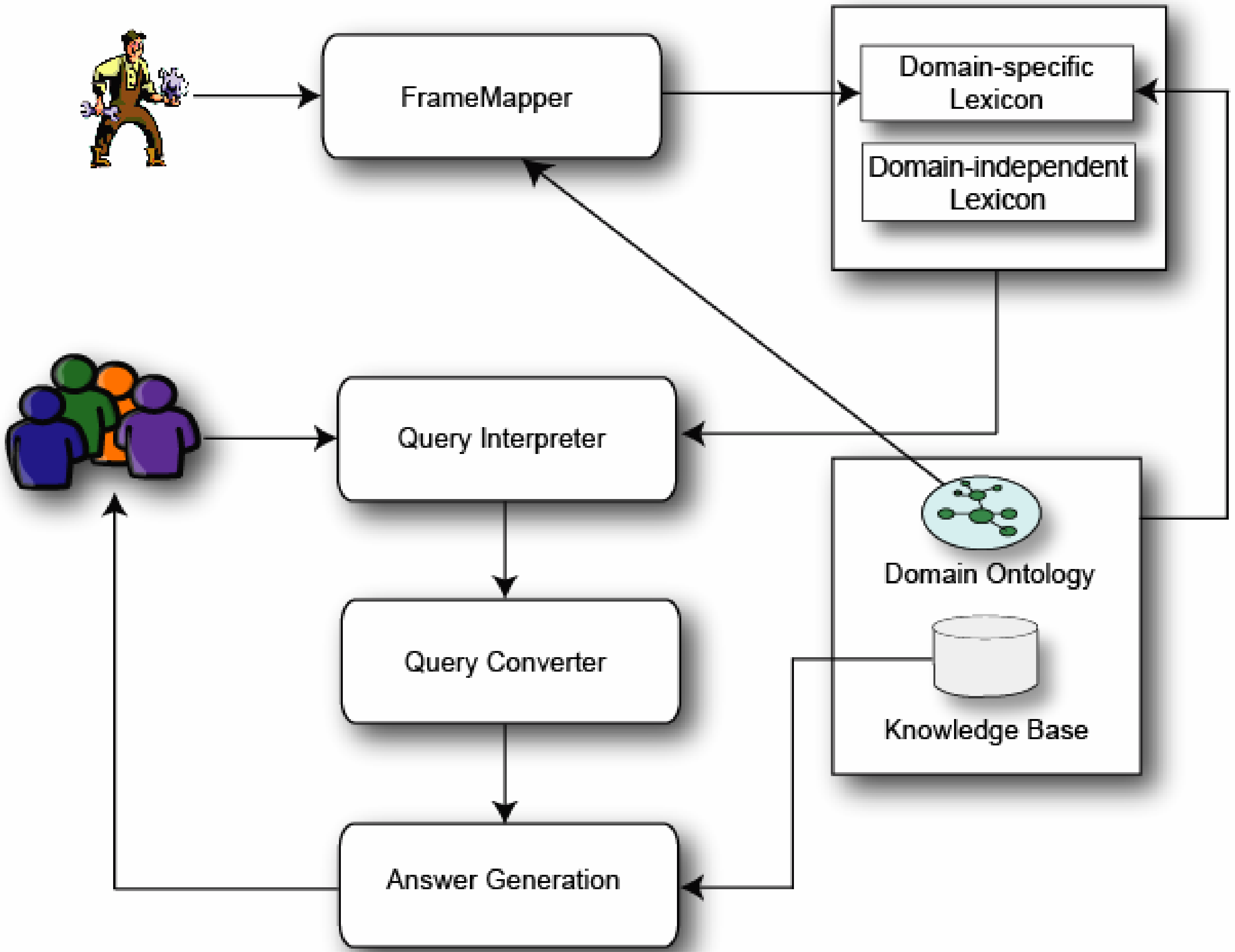
- Definition: tool allowing users to query a knowledge base using (restricted/unrestricted) natural language

- Challange: Translation of natural language queries into formal, structured queries

Which river flows through more cities than the Rhein?

$$\forall W \leftarrow W : river \wedge \exists V_1, V_2, V_3, V_4\ (V_1 : city \wedge rhein[flowThrough \rightarrow V_1] \wedge W[flowThrough \rightarrow V_2] \wedge V_2 : city \wedge count(W, V_2, V_3) \wedge count(\_, V_1, V_4) \wedge greater(V_3, V_4))$$

The ORAKEL System

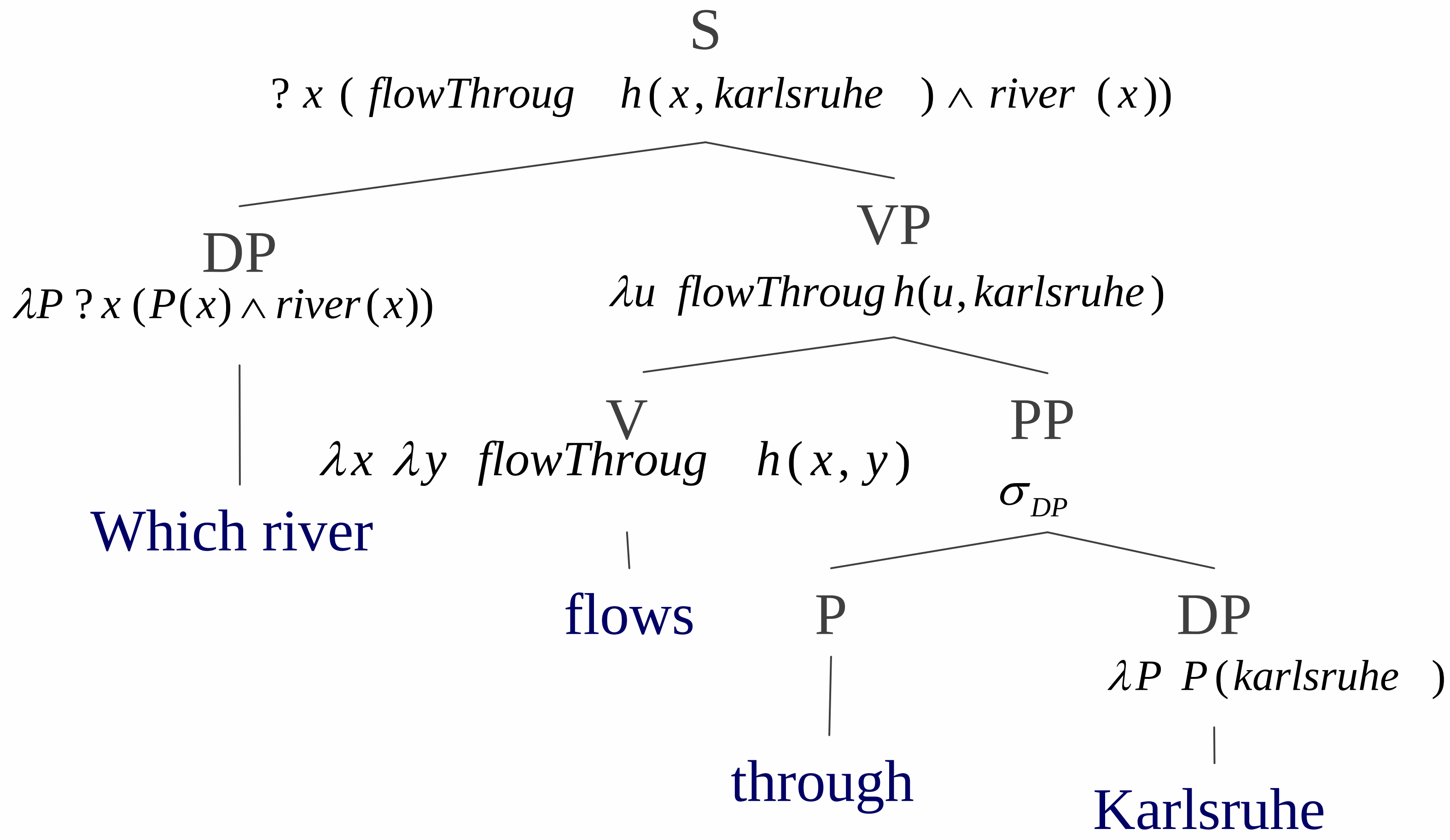


The ORAKEL System

- ORAKEL is a natural language interface implementing a standard syntax-semantics interface
- Standard compositional semantics approach, i.e. the meaning of a query is composed of the meanings of the words and the way they are connected
- Parse tree is used to guide the incremental semantics composition
- Meaning is captured through lambda expressions

- It requires a rich lexicon mapping linguistic expressions to predicates defined in the ontology

Query Interpreter – Meaning Construction



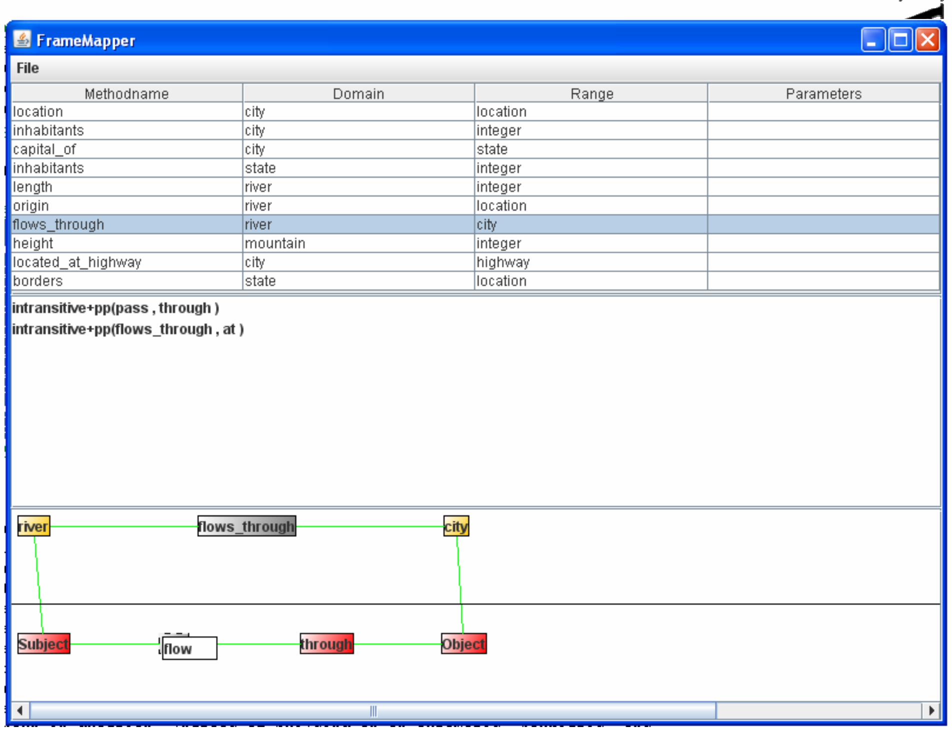
Bi-partite Ontology Lexicon

- Domain-independent Lexicon
  - Contains closed-class words with constant meaning across domains:
    - Determiners: every, most, the most, a, the only, the, all, no, …
    - Prepositions: after, before, in (spatial), in (temporal), …
    - Question pronouns: who, what, which, when, where, …
  - Meaning is captured with respect to foundational categories, e.g. as provided by DOLCE (no manual work by user)

- Domain-specific Lexicon
  - Contains lexical representation of instances and concepts, and relations
  - Partially generated automatically from the ontology, relying on its labels, partially created by the user
  - Lexicon used to generate syntactic variants, e.g. plural forms

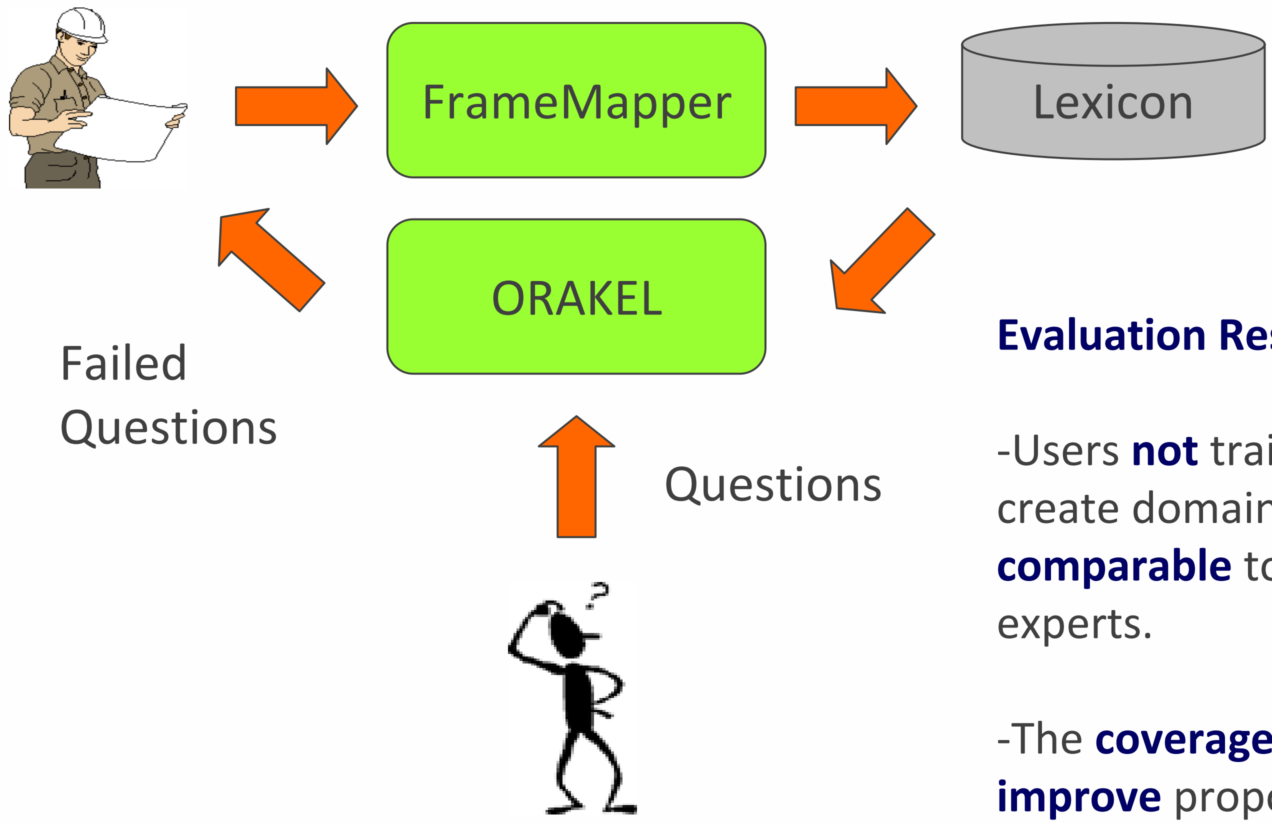
Domain-specific lexicon
Adaptation with FrameMapper

- Subcategorization Frames: linguistic predicate-argument stuctures
  - flow(subj,pcomp(through))
- Relations in the Knowledge Base:
  - flows_through(river,city)
- Basic idea: user performs mapping between arguments of a subcategorization frame and a relation in the knowledge base
- Domain-specific lexicon is generated in the background as a byproduct of the mappings performed by a lexicon engineer



Benefit: Users **without familiarity** with computational linguistics can customize the system to work with a specific knowledge base

Adaptation Methodology



Evaluation Results

-Users **not** trained in NLP are able to create domain-specific lexica **comparable** to those created by NLP experts.

-The **coverage** of the lexicon will **improve** proportionally to the number of iterations performed.

Application Example: The BT Digital Library

- EU-IST IP SEKT "Semantically Enabled Knowledge Technologies" (2004 – 2006)
- Case study: digital library at British Telecom (BT)
  - 8000 users (2500 regular) vs. 5 million documents
  - heterogeneous content
  - limited capabilities of existing user interface
- Goal: show the potential of semantic technologies in a digital library scenario

Scenario (BT Digital Library)

Bob works as technology analyst for British Telecom. His daily work includes research on new technological trends, market developments as well as the analysis of competitors.

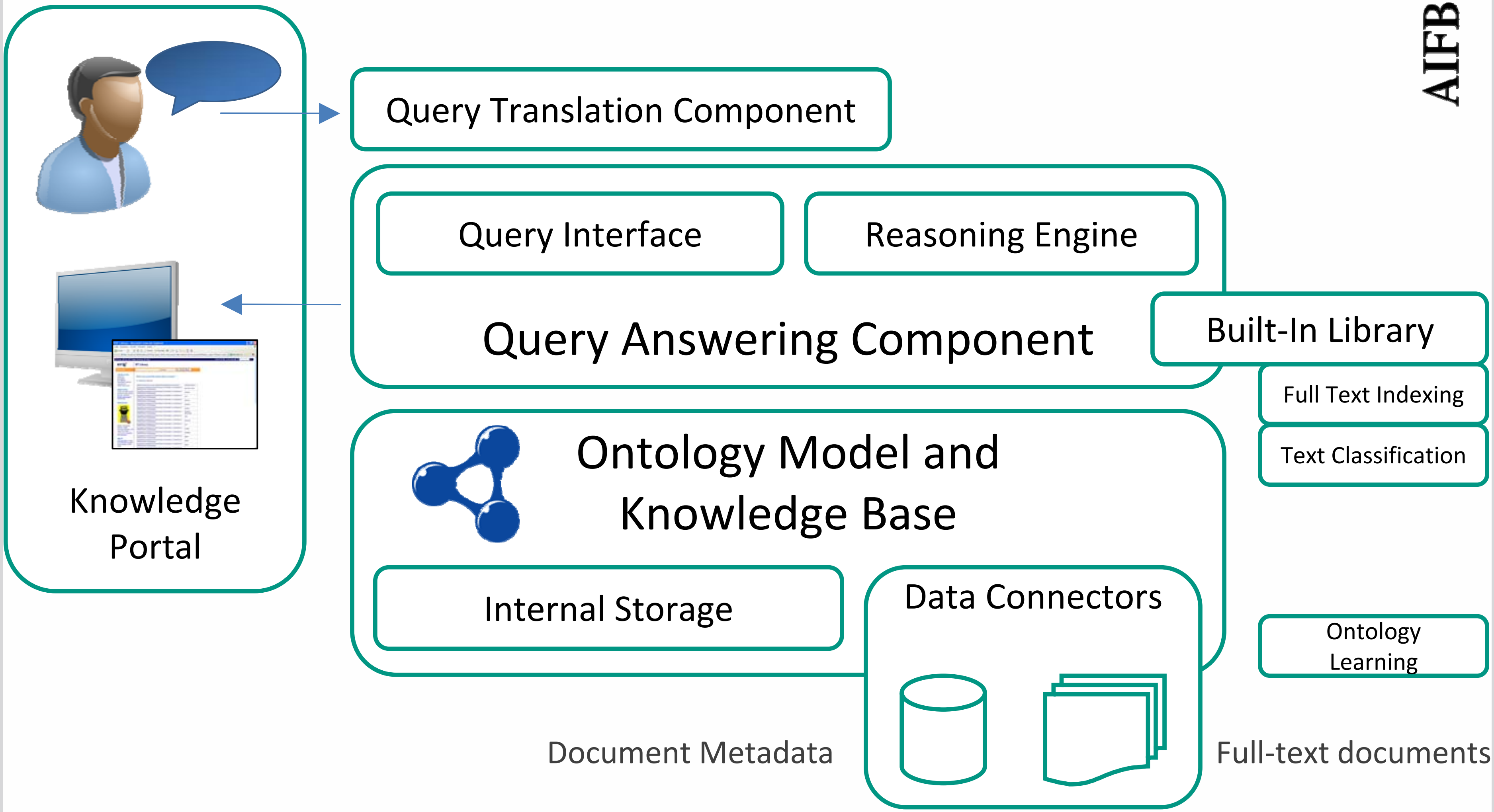
Bob's company maintains a digital library that gives access to a repository of internal surveys and analysis documents. The company also has a license with an academic research database which is accessed via a separate interface.

Depending on his work context, Bob uses the topic hierarchies, the full-text search functionalities or metadata search facilities provided by the two libraries to get access to the relevant data.

However, Bob is often annoyed by the differing topic hierarchies and metadata schemes used by the two libraries as well as by a cumbersome syntax for metadata queries.

- Heterogeneity of content
- Heterogeneity of search facilities
- Heterogeneity of data models (schemas)
- Interface design challenge

Conceptual Architecture



Ontology Model and Knowledge Base

- Ontology (PROTON top level ontology)
  - global conceptual model
  - aligned with established schemas (e.g. Dublin Core)
  - expressed by means of W3C standards (OWL/RDF)
- Knowledge base of the digital library
  - actual bibliographical metadata, topic hierarchies, and full-text document content
  - data aligned with global ontology via mapping axioms

```
swrc:Book             rdfs:subClassOf     protont:Document
expl:document5127     rdf:type            swrc:InProceedings
expl:document5127     protont:title       "Digital Libraries"
```

Query Answering Component

- Management of ontologies (KAON2 infrastructure)
- Query answering against knowledge base (SPARQL)

```
SELECT ?x WHERE {
    ?x rdf:type <http://proton.semanticweb.org/2005/04/protonu#Article> .
    ?x <http://proton.semanticweb.org/2005/04/protont#hasSubject> ?y .
    ?y rdfs:label ?z .
    match(?z,"Intellectual Capital")
}
```

- Transparent steps during query answering:
  - reasoning engine draws implicit inferences
  - some predicate evaluations "pushed-down" to data sources
  - some predicate evaluations "pushed-down" to "built-ins

Preprocessing and Extraction: Integrating Unstructured Content

- Full-text indexing via built-in
- Text classification via built-in
  - useful for emerging or user-specified topics
  - on-the-fly evaluation via built-in
- Ontology learning component (Text2Onto)
  - extracts ontological primitives from textual content
  - state-of-the-art NLP and text mining techniques

Knowledge Portal

- Interaction via standard interfaces
  - keyword-search, topic browsers etc.
- Interaction by asking natural language queries

" Who wrote books on 'digital libraries'?"
"Which journal articles were written by 'Tim Berners-Lee' (and for which journal)?"

Natural Language Interface

- Query translation component (ORAKEL)
  - converts natural language queries into SPARQL queries
  - queries are evaluated against knowledge base
- Translation step comprises
  - deep parsing of the questions
  - roughly, linguistic frames become query constraints
  - lexicon describes possible lexical realizations of ontology elements

[P. Cimiano, P. Haase, J. Heizmann: "Porting Natural Language Interfaces between Domains - An Experimental User Study with the ORAKEL System", ICIUI, 2007]

Scenario Revisited

“Which journal articles were written by 'Tim Berners-Lee' for which journal?”

```
PREFIX protonu:<http://proton.semanticweb.org/2005/04/protonu#>
PREFIX protont:<http://proton.semanticweb.org/2005/04/protont#>

SELECT ?x ?z WHERE {
   ?x rdf:type protonu:Article .
   ?x protont:documentAuthor ?y .
   ?y rdfs:label ?ys .
   match(?ys, "Tim Berners Lee") .
   ?z rdf:type protonu:Journal .
   ?x protonu:publishedWithin ?z
}
```

```
"The Semantic Web"                  "The Scientific American"
"WWW: Past, Present, and Future"   "IEEE Computer"
[...]                               [...]
```

Scenario Revisited

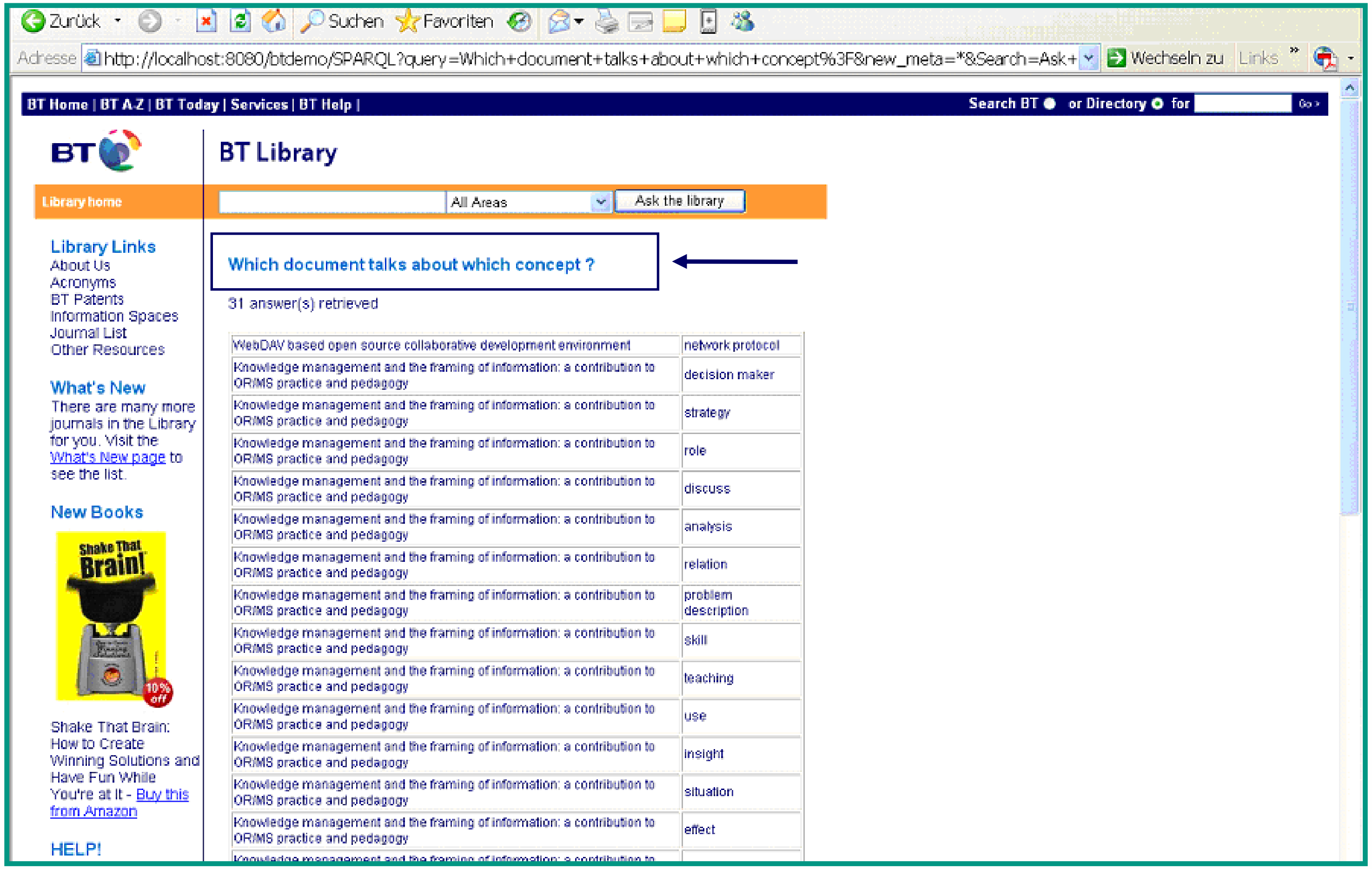
“Who wrote which book classified as 'digital libraries'?"

```
PREFIX protonu:<http://proton.semanticweb.org/2005/04/protonu#>
PREFIX protont:<http://proton.semanticweb.org/2005/04/protont#>

SELECT ?x ?y WHERE {
  ?y rdf:type protonu:Book .
  ?x protont:documentAuthor ?y .
  ?y protont:documentAbstract ?z .
  EVALUATE ?margin:=classify(?z,'digital_libraries') .
  FILTER ?margin>0
}
```

```
"Digital Libraries"                  "William Y. Arms"
"Understanding Digital Libraries"    "Michael Lesk"
"How to Build a Digital Library"     "Ian Witten"
[...]                                [...]
```

Example: The BT Digital Library (cont.)



Screenshot from BT Digital Library

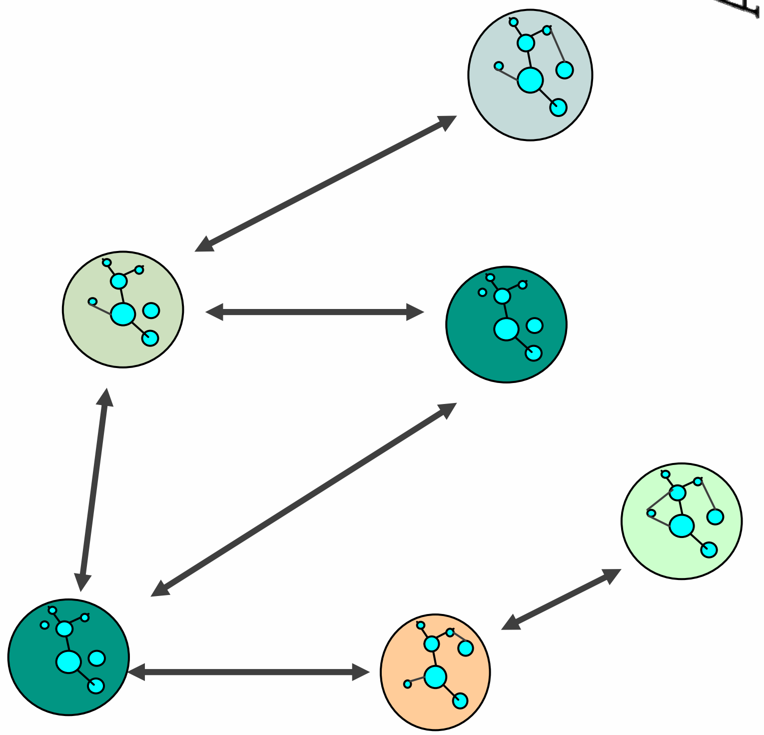
Topics

- Semantic Search
  - Overview
  - Ontology-based Information Retrieval
  - Ontology-based Query Interpretation
  - Natural Language Interfaces
  - Architectural Aspects and Examples
- **Information Integration**
  - Ontology Mapping
  - Automated Mapping Discovery

Ontology Mapping – Problem and Scope

- The Problem
  - ***Heterogeneous ontologies require mappings for interoperability***
  - Numerous independent Ontologies
  - No single Domain Model
  - Modeling same or overlapping Knowledge

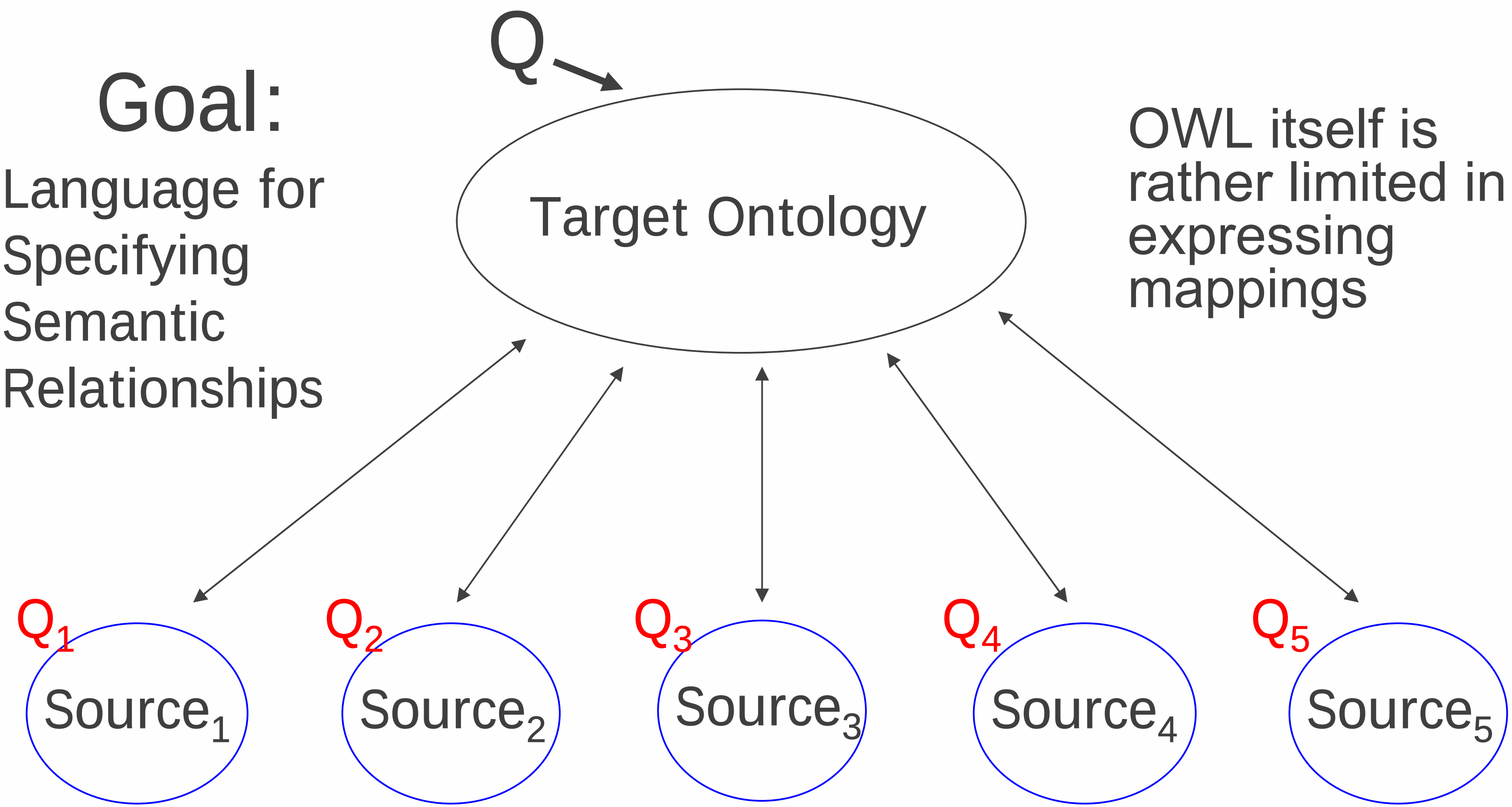
- Main challenges
  - Identifying mappings (correspondences between Entities)
  - Representing these Relations
  - Utilizing Mapping for querying, reasoning, ontology integration, translation and exchange



Mapping Systems for Ontology Integration

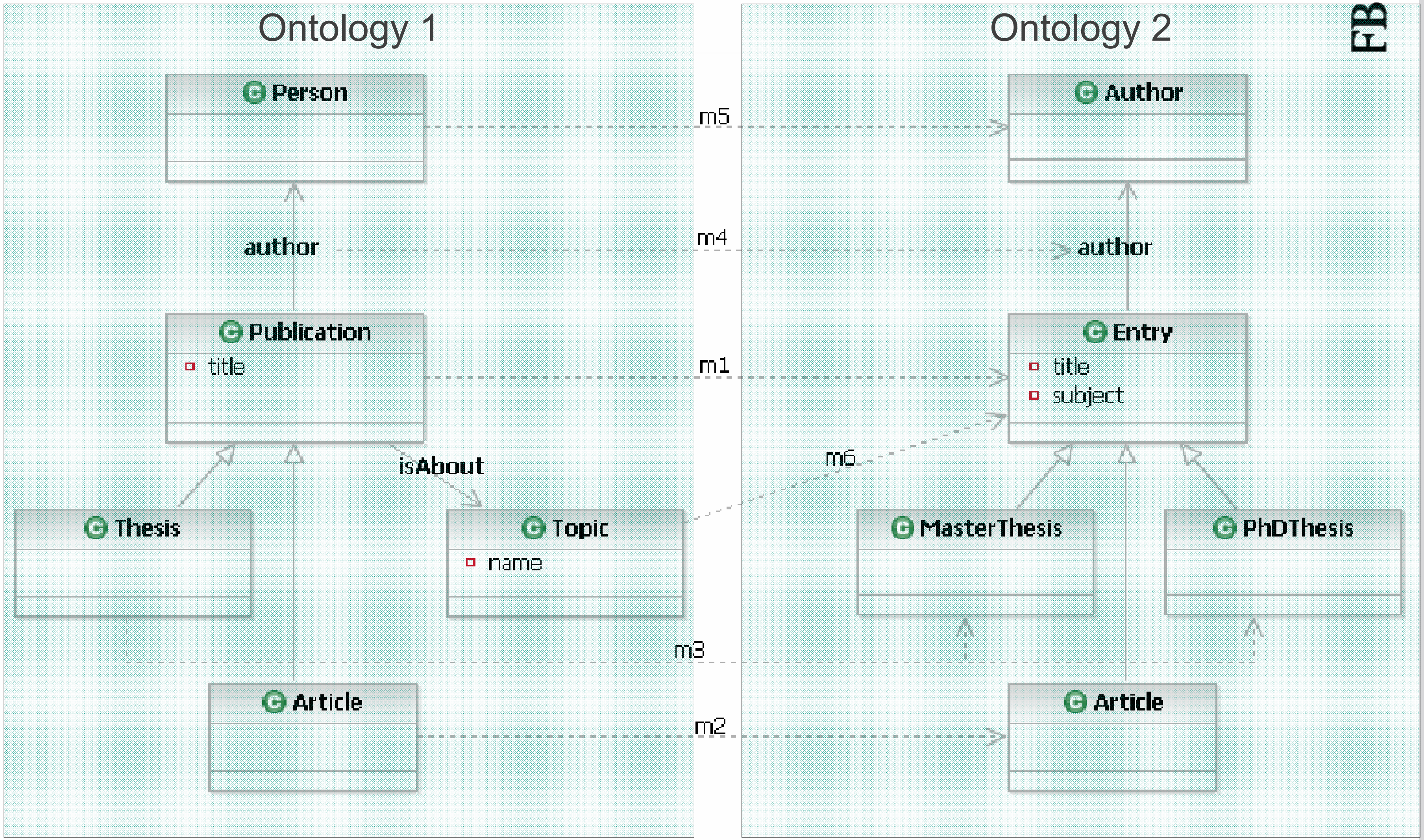
Goal:

Language for Specifying Semantic Relationships



OWL itself is rather limited in expressing mappings

Sample Mapping



OWL DL Mapping System

- An OWL DL mapping system is a triple ($\mathcal{S}$, $\mathcal{T}$, $\mathcal{M}$), where
  - $\mathcal{S}$ is the source OWL DL ontology
  - $\mathcal{T}$ is the target OWL DL ontology
  - $\mathcal{M}$ is the mapping between $\mathcal{S}$ and $\mathcal{T}$
- Mapping: set of assertions
  - $q_S \sqsubseteq q_T$ (sound mapping)
  - $q_S \sqsupseteq q_T$ (complete mapping)
  - $q_S \equiv q_T$ (exact mapping)
  - where q_S and q_T are conjunctive queries over $\mathcal{S}$ and $\mathcal{T}$, respectively, with the same set of distinguished variables
- Semantics defined via translation into FOL, computing answers against $\mathcal{S} \cup \mathcal{T} \cup \mathcal{M}$

[Haase and Motik, IHIS05]

Examples

- Correspondences between atomic elements
  - s: Publication(x) $\sqsubseteq$ t:Entry(x)
  - s:author(x,y) $\sqsubseteq$ t:author(x,y)
- Correspondences between complex class descriptions
  - s:Thesis(x) $\sqsubseteq$ t:PhDThesis$\sqcup$t:MasterThesis(x)
- Even more complex mappings
  - s: Publication(x)$\wedge$ isAbout(x,y) $\wedge$ name(y,z) $\sqsubseteq$ t:Entry(x)$\wedge$ subject(x,z)

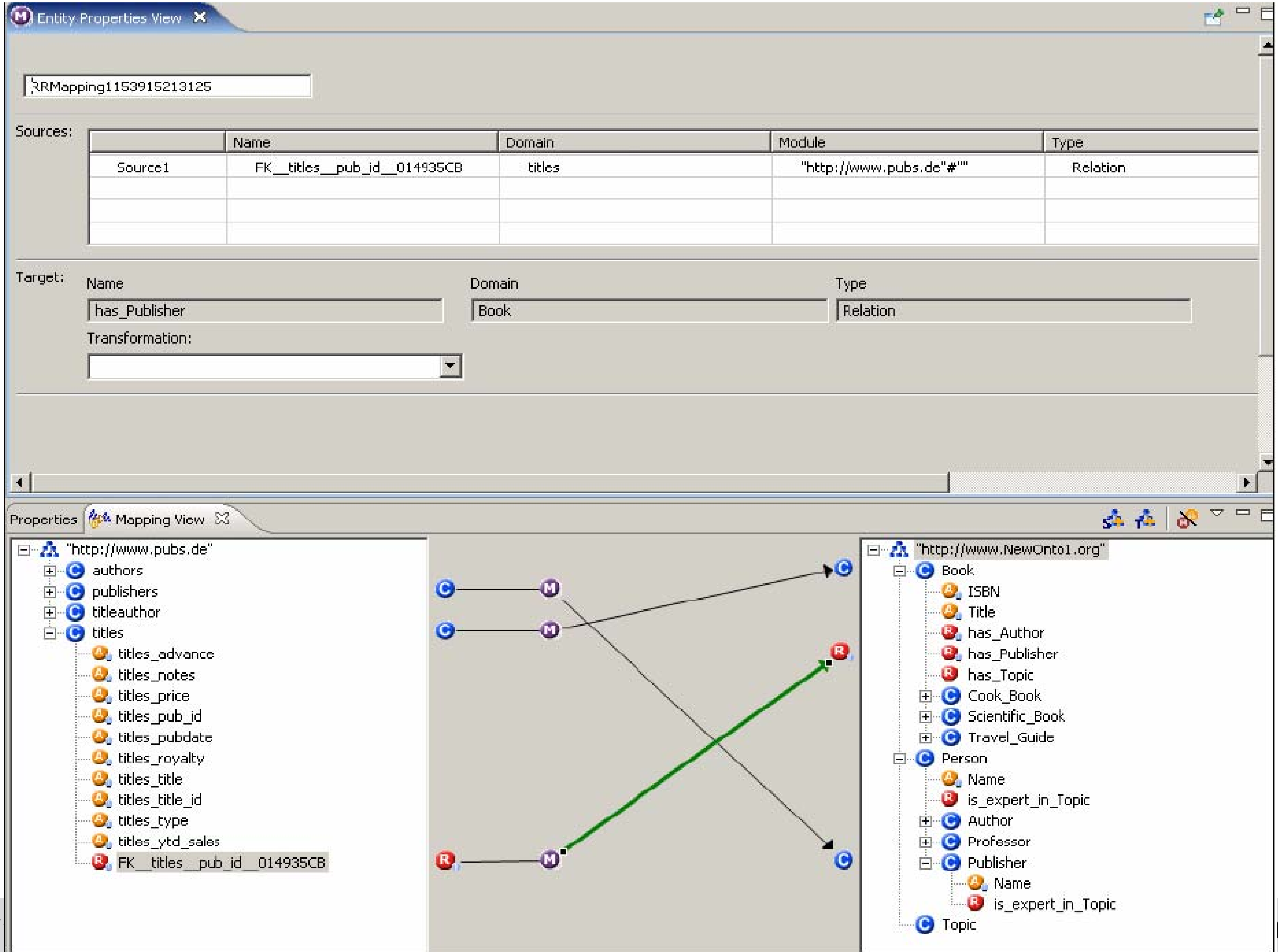
Ontology Mapping – Techniques and Tools

- Great number of Techniques
  - Syntactic, Semantic, External
  - Element-Level, Structure-Level
  - Schema or Instance Level mapping

- Mapping Tools
  - Several mapping systems already available
    (GLUE, PROMPT, FOAM, ONION, MAFRA)
  - Manual, visual creation of mappings between ontologies
  - Integration of (relational databases): automated ontology lifting and query answering
    (OntoMap, ODEMapster)

- Best results
  - Find best approximate Matches -> Similarity
  - Semi–automatic
  - Requires human Domain Expert

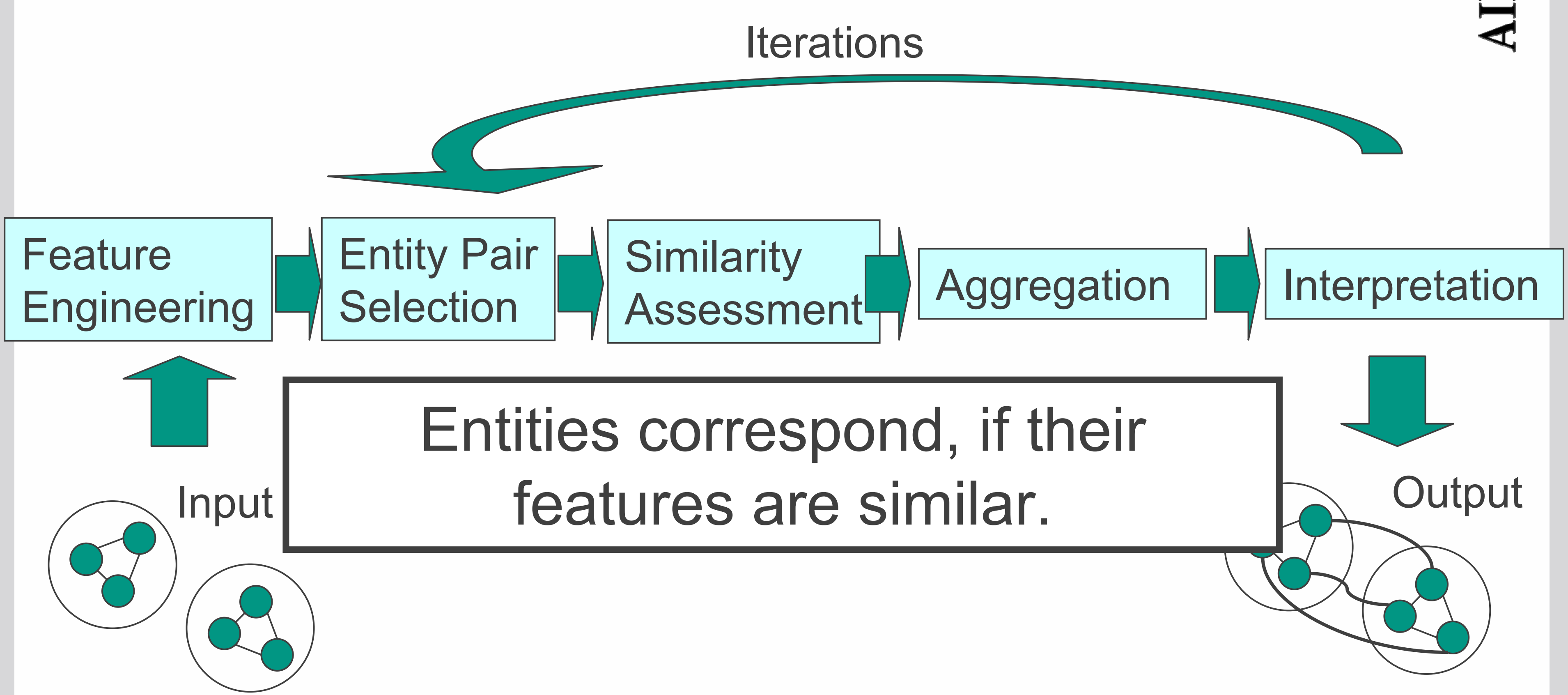
Ontology Mapping with OntoMap



Topics

- Semantic Search
  - Overview
  - Ontology-based Information Retrieval
  - Ontology-based Query Interpretation
  - Natural Language Interfaces
  - Architectural Aspects and Examples
- Information Integration
  - Ontology Mapping
  - **Automated Mapping Discovery**

Automated Mapping Discovery Process



Features and Similarity Measures

	Feature	*Similarity Measure*
Concepts	label	String Similarity
	subclassOf / superclassOf	Set Similarity
	instances	Set Similarity
	…	
Relations	Domain, Range	Set Similarity
	…	
Instances		

Similarity Measures

- String similarity

$$sim_{String}(s_1, s_2) = \max(0, \frac{\min(|s_1|, |s_2|) - ed(s_1, s_2)}{\min(|s_1|, |s_2|)})$$

- Set similarity

$$sim_{Set}(S_1, S_2) = \underset{e_i \in S_i}{avg} \max_{e_j \in S_j}(sim(e_i, e_j))$$

Aggregation of multiple similarity measures

- Weighted combination method
  - Manually
  - Machine learning

$$sim(e,f) = \sum_k w_k sim_k(e,f)$$

- Non-weighted combination method
  - Average
  - Maximal
  - Minimal

- OWA – Ordered Weighted Average