Ontology Lifecycle

Ontology

„People can’t share knowledge if they do not speak a common language.“ [Davenport & Prusak, 1998]

„An ontology is an explicit specification of a conceptualization.“ [Gruber, 1993]

- Ontologies enable a better communication between Humans/Machines
- Ontologies standardize and formalize the meaning of words through concepts

Ontology & Metadata

Explicit vs. Implicit Knowledge

Socialisation

Implicit Knowledge

From

Explicit Knowledge

Externalisation

Implicit Knowledge

Internalisation

Explicit Knowledge

Combination / Integration
Case study: OntoWeb.org

Portal Generation
Navigation
Query/Search
Content
Integration
Collect metadata from participating partners
Annotation

Ontology-based Processes

Knowledge Meta Process
Design, Implementation, Evolution of Ontology

Knowledge Process
Usage of Ontology

OTK Methodology: Knowledge Meta Process

- **Task**: Build ontology based KM applications
- **Problems**:
  - Collaboration between domain experts and knowledge engineers
  - Evaluation of ontologies

- Process-oriented, cyclic
- Pre-defined decisions and outcomes for each step
- Links to further existing methodologies for substeps

OTK Methodology: Knowledge Meta Process

- Feasibility study
- Kickoff
- Refinement
- Evaluation
- Application & Evolution

Knowledge Management Application

Human Issues
Software Engineering

Ontology Development
OTK Methodology: Knowledge Meta Process

- Process-oriented, cyclic
- Pre-defined decisions and outcomes for each step
- Links to further existing methodologies for substeps

Tools

- OntoKick: Capture Requirements Specification
- Mind2Onto: Brainstorming
- OntoFiller: Documentation & Translation
- OntoClean: Formal Ontology Evaluation
- SesamePlugin: Storage & Versioning

Feasibility Study

- KM systems only function satisfactorily if they are properly integrated into the organization
- Many factors other than technology determine the success of such a system
- (Based on CommonKADS)

- Focus domain for ontology
- Identify people involved
- GO / No GO decision
Feasibility study

Current State: Skills Management

• Employee data distributed over many systems
• Different schemata for data
• Incomplete data

Feasibility study

Intended state: Skills Management

Expert search

Knowledge gap analysis

Personal development

Intellectual Capital Assessment

OTK Methodology:
Knowledge Meta Process

Ontology Kickoff

• Ontology Requirements Specification Document (ORSD)
  1. Domain & Goal
  2. Design guidelines
  3. Available knowledge sources
  4. Potential users and user scenarios
  5. Applications supported by the ontology

• Analyze knowledge sources
• Develop baseline ontology description

Draft version, typically most important concepts and relations are identified and described as an untyped graph
ORS – Ontology Requirements Specification

- **Goal of the ontology:**
  - Tracking and analyzing corporate business histories
- **Domain and Scope:**
  - Merger & acquisition, restructuring, management changes and other strategic activities in the chemical industry
- **Supported Applications:**
  - Web-based Corporate History Analyzer
- **Knowledge Sources:**
  - Research analysts (domain experts)
  - Document: c:/mydocuments/superdokument.doc
  - URL: http://www.webpage.com
- **Users and Use Cases:**
  - Users: Research analysts, strategic consultants
  - Use Case 1: Track strategies of specific companies
  - Use Case 2: Analyze strategic moves of competitors
- **Competency Questions:**
  - Attached Competency Questionnaire
  - Potentially reusable ontologies:
    - not known

---

CQ – Competency Questionnaire

<table>
<thead>
<tr>
<th>CQ Nr.</th>
<th>Competency Question</th>
<th>Concepts</th>
<th>Relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQ1</td>
<td>What are the subsidiaries, divisions and locations of company X?</td>
<td>company, subsidiary, division, location</td>
<td>company has subsidiary company has division company has location</td>
</tr>
<tr>
<td>CQ2</td>
<td>Which companies acquired company X?</td>
<td>company, acquisition</td>
<td>company makes acquisition acquisition has buyer acquisition has seller</td>
</tr>
<tr>
<td>CQ3</td>
<td>Which companies merged in 1990 in the rubber industry?</td>
<td>company, merger, year, industry</td>
<td>company makes merger company isPartOf industry merger happensIn year</td>
</tr>
<tr>
<td>CQ4</td>
<td>Who is CEO of company X?</td>
<td>CEO, company</td>
<td>company has CEO</td>
</tr>
</tbody>
</table>

---

Kick-Off

- Ontology workshop to train domain experts in ontology modelling for
  - IT
  - Private customer insurance
  - Human Resource Management
- First version of domain ontology by expert
  - Manual development of ontology
  - Brainstorming (Mind Maps)
  - Middle-out approach
- Result: approx 700 Concepts in about 4 weeks
Requirement specification

Knowledge Sources

Design Guidelines
Knowledge Sources

Competency questions

Competency questions

Competency questions
Traceability

Mind2Onto

• Task: Collaborative capturing of domain knowledge through domain experts and modelling experts

• Problem: Collaboration with domain experts who have:
  – No experience with modelling
  – No time for modelling
• Task: Collaborative capturing of domain knowledge through domain experts and modelling experts

• Problem: Collaboration with domain experts who have:
  – No experience with modelling
  – No time for modelling

Mind2Onto

MindManager: Standard software for the creation of electronic MindMaps

Advantage: Intuitive, understandable
Problem: Semantics of MindMaps only vaguely defined

Export to OntoEdit

OntoEdit/OntoFiller

OntoFiller: Support for translation and documentation of concepts and relations in multiple languages

OtK Methodology: Knowledge Meta Process

Refinement

• Knowledge elicitation with domain experts
  – Refine concepts and relations
  – Typically axioms are identified

• Formalize
  – E.g. F-Logic, DAML+OIL
  – Axioms depend on language capabilities

• Develop and refine ontology
Inferencing

Theoretical Issues
- F-Logic
  - Object-oriented
  - Deductive Database-oriented
  - Well-founded semantics

Practical Issues
- Namespace mechanism: Ontologies/Ontology Parts -> modules
- Switch-off definitions:
  - For testing
  - For fast executions without consistency checks
- DB Connectors: map DB tables via JDBC
- User-definable built-ins
- Extensive API:
  - remotely connect to the inference engine
  - import and export several standards (e.g., RDF(S))

Exploit Inferencing

- Hook in existing resources with inferencing
  - Jdbc
  - Rules
- Construct axiom libraries
  - Temporal reasoning
  - PartWhole reasoning
  - ...
- Selective axiom applications
  - F-Logic semantics: E.g. type coercion at concept level
  - Domain specific consistency: non-cyclic hasPart
  - Axioms for modeling policies
  - Debugging

Contrast: OilEd

Steffen Staab
ISWeb - Lecture „Semantic Web“ (43)
OTK Methodology: Knowledge Meta Process

- Feasibility study
- Kickoff
- Refinement
- Evaluation
- Application & Evolution

Knowledge Management Application

Human Issues

Software Engineering

Evaluation

- Check requirements (ORSD)
  - Are all CQs answered?
  - Is the ontology within the scope?

- Test in target application
  - Analyze usage patterns

- Deploy application(s)

OntoClean

- Task: Formal evaluation of ontologies

- Well-known methodology:
  OntoClean [Welty & Guarino, 2001]
  - Aims at „cleaning“ of hierarchies
  - Based on philosophical notions
    - „essence“, „rigidity“, „identity“, „unity“ ...
    - etc.

- Implementations: For F-Logic & OWL

OntoClean: Definitions

„Essence“: A property is essential for an individual iff. it necessarily holds for that individual.

Example: York is necessarily a person.

„Rigidity“
- A property is „rigid“ (+R) iff. it is necessarily essential for all its individuals.
- A property is „non-rigid“ (-R) iff. it is not essential for some of its individuals.
- A property is „anti-rigid“ (~R) iff. it is not essential for all its individuals.

Example: „Person“ is necessarily an essential property for all its individuals.

- There exist similar definitions for „identity“ (+I, -I, +O, -O), „unity“ (+U, -U, ~U), „dependency“ (+D, -D), ... etc. ...
OntoClean: Classification & ideal structure

| +O +1 +R +D | Type       |
| +O +1 +R -D | Quasitype  |
| -O +1 -R +D | Material role |
| -O +1 -R -D | Phased sortal |
| -O -1 -R +D | Mixin      |
| -O -1 -R -D | Category   |
| -O -1 -R +D | Formal Role |
| -O -1 -R -D | Attribution |
| +O -1 -R -D | Incoherent |

See: [Welty & Guarino, 2001]

OntoClean: Layering

- D +O +R +U

OntoCleanPlugin: Formalisation of meta ontology
OntoCleanPlugin: Formalisation of meta ontology

Uppermost concept "Property" of the meta ontology has attached all relations necessary for classifying concepts of an ontology.

OntoCleanPlugin: Formalisation of axioms

- Anti-rigid concepts (~R) cannot have rigid subconcepts (+R)
- Etc.

Cleaning example

Def.: Being an active participant in some event.
OntoCleanPlugin: Cleaning example

Evaluating the query \( \forall Y, X, Y, Z \rightarrow \text{check}(Y, X, Y, Z) \)

\[ \text{Error: Agent } \neg \text{R} \text{ can't subsume Person } \neg \text{R} \]

Person should not be a subconcept of Agent!
Interpretation: Persons can be agents, but persons are not necessarily agents.

„Is York an agent?“

OTK Methodology: Knowledge Meta Process

Worksheet for life cycle aspects of ontology

- Who is going to maintain it?
- Who is going to pay for it?
- What is the resulting quality (increase, decrease)?
- How large are the network costs (cost of negotiation grows quadratic with number of participants)?
- What is the expected life time of the ontology?
- How brittle is it with regard to updates?
- What error types will occur/are relevant?
Worksheet for life cycle aspects of metadata

- ala ontology
- Co-ordinated change of data and metadata?
- Co-ordinated change of ontology and metadata?

Rule of thumb – costs:
- Hardware 1
- Software 10
- Daten 100

- Cold start (chicken-and-egg) problem: A problem? How to overcome?
- Granularity of metadata envisaged: classification, identification of people/events/relationship s/etc.

Coordination of metadata & ontology

- Match or mismatch between the two,
  - E.g. classification only, but ontology about transitive relationships

Type-1 Error

- False Positive
  - Often dominating problem in company internal IR
  - It can be more costly to learn about all low-price provider of pens than to just select from a sample

Type-2 Error

- False negative: Positive example not detected as such
  - Often not critical for information retrieval
    - „show me bookstores who sell the `CommonKADS' book“
  - Often critical for B2B operations
    - „whether `6000 computer' is mapped to `IBM RS/6000 SP system' or to `HP OmniBook Laptop 6000' is a large difference with regard to price and performance“
Refined Error types (Halo Project)

- 1. (MOD) Knowledge Modeling: the ability of the knowledge engineer to model information/write axioms
- 2. (IMP) Knowledge Implementation/Modeling Language: the ability of the representation language to accurately represent axioms
- 3. (INF) Inference and Reasoning: the ability of the inference engine to "find the needle in the haystack"
- 4. (KFL) Knowledge Formation and Learning: the ability of the system (KB + inference engine) to acquire and merge knowledge through automated and semi-automated techniques
- 5. (SCL) Scalability: the ability of the KB to scale

- http://www.haloproject.com

Refined Error types II (Halo Project)

- 6. (MGT) Knowledge Management: the ability of the system to maintain, track changes, test, organize, document; the ability of the knowledge engineer to search for knowledge
- 7. (OMN) Query Management: the ability of the system to robustly answer queries
- 8. (ANJ) Answer Justification: the ability of the system to provide justifications for answers in the correct context and resolution
- 9. (QMT) Quality Metrics: the ability of the developers to determine how "good" the knowledge base is at any given point in its evolution
- 10. (MTA) Meta Capabilities: the system's ability to utilize meta-reasoning or meta-knowledge

Ontology Evolution: Technical aspects

- Ontology development is necessarily an iterative and a dynamic process
- Ontologies must be able to evolve for a number of reasons:
  - Application domains and user's needs are changing
  - System can be improved
- Developing ontologies is expensive, but evolving them is even more expensive

Requirements for ontology evolution

- Functional requirement:
  - enable the handling of the required changes
  - ensure the consistency of the underlying ontology and all dependent artifacts
- Interaction requirement – supports the user to manage changes more easily
- Refinement requirement – offers advice to the user for continual system refinement
### Ontology Evolution – Change representation

<table>
<thead>
<tr>
<th>Composite change</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move concept</td>
<td>Move a concept from one parent to another.</td>
</tr>
<tr>
<td>Merge concepts</td>
<td>Replace several concepts with one and aggregate all instances.</td>
</tr>
<tr>
<td>Extract subconcepts</td>
<td>Split a concept into several subconcepts and distribute properties among them.</td>
</tr>
<tr>
<td>Extract superconcept</td>
<td>Create a common superconcept for a set of unrelated concepts and transfer common properties to it.</td>
</tr>
<tr>
<td>Extract related concept</td>
<td>Extract related information into a new concept and relate it to the original concept.</td>
</tr>
<tr>
<td>Shallow concept copy</td>
<td>Duplicate a concept with all its properties.</td>
</tr>
<tr>
<td>Deep concept copy</td>
<td>Recursively apply shallow copy to all subconcepts of a concept.</td>
</tr>
<tr>
<td>Pull up properties</td>
<td>Move properties from a subconcept to a superconcept.</td>
</tr>
<tr>
<td>Pull down properties</td>
<td>Move properties from a superconcept to a subconcept.</td>
</tr>
<tr>
<td>Move properties</td>
<td>Move properties from one concept to another concept.</td>
</tr>
<tr>
<td>Shallow property copy</td>
<td>Duplicate a property with same domain and range.</td>
</tr>
<tr>
<td>Deep property copy</td>
<td>Recursively apply shallow copy to all subproperties of a property.</td>
</tr>
<tr>
<td>Move Instance</td>
<td>Moves an instance from one concept to another.</td>
</tr>
</tbody>
</table>

- **Composite changes**
  - They are more powerful
  - They have coarser granularity
  - They have often more meaningful semantics

- **Elementary changes**
  - They can not be decomposed into simpler ones
  - They heavily depend on the underlying ontology model

\[
\text{MoveConcept} \neq (\text{RemoveSubConcept} + \text{AddSubConcept})
\]
Ontology Evolution – Semantics of change

- Enables resolution of changes in a systematic manner, ensuring consistency of the whole ontology

Ontology Evolution – Change implementation

- After user’s approval all changes are applied to the ontology
- Since it is necessary to perform several changes together, the transaction server is needed.

Evolution Strategies

An evolution strategy unambiguously defines the way how changes will be resolved

Resolution points:
- how to handle orphaned concepts;
- how to handle orphaned properties;
- how to propagate properties to the concept whose parent changes;
- what constitutes a valid domain of a property;
- what constitutes a valid range of a property;
- whether a domain (range) of a property can contain a concept that is at the same time a subconcept of some other domain (range) concept;
- the allowed shape of the concept hierarchy;
- the allowed shape of the property hierarchy;
- ...

Elementary evolution strategies

- delete
- reconnect to the root
- reconnect to the superconcepts

Common policy consisting of a set of elementary evolution strategies, each giving an answer for one resolution point, is an evolution strategy.
Advanced evolution strategies

Mechanism to prioritize and arbitrate among different evolution strategies, relieving the user of choosing them individually:

- structure-driven strategy
- process-driven strategy
- instance-driven strategy
- frequency-driven strategy

Implementation

Applications & Services
- OIModeler - Ontology and Metadata Engineering Tool
- KAON Portal and other User Interface Applications and Services

Middleware
- Evolution Strategy
- Reversibility Services
- Evolution Logging

KAON Access Interface
- KAON API
- RDF API
- KAON RDF Server

Data and Remote Services
- Persistence, Transactions, Security
Evolution wrap-up

OntoLogging:
- process-based approach for ontology evolution
- Evolution strategies that enable the customisation of the ontology evolution process
- Implementation in KAON framework

Ongoing work:
- Evolution between distributed ontologies
- Change discovery

OTK Methodology: Knowledge Meta Process

Conclusions on Knowledge Meta Process
Experiences from OTK Case Studies

- **Guidelines** for domain experts from industry have to be pragmatic
  1. Train the user about ontologies
  2. Show the concrete advantage of the KMS
  3. Model precisely – but allow for imprecise views (most users cannot distinguish classes vs instances or isa vs partOf)

- **Plan for Maintenance**
- **Avoid/Reduce chicken-and-egg problem**
  1. Plan für content that makes KMS interesting
  2. Show quick win

- **Collaborative ontology engineering** requires sophisticated tool support and physical presence
- **Brainstorming** is a valuable add-on during the early stages of ontology engineering

Knowledge Process

- **Apply**
- **Summarize**
- **Analyze**
- **Use**
- **Creation**
- **Import**
- **Retrieval/Access**
- **Capture**
- **Annotate**
- **Extract**
- **Search**
- **Query**
- **Inferencing**
- **Generate**
- **Views**

OTK Case Study @ BT

- **Ontology Creation**
- **Import**
- **Capture**
- **Annotate**
- **Extract**
- **Metadata**
- **Documents**

Users Portal

- **OntoShare**
- **OntoExtract**
- **OntoWrapper**
- **Sesame**
- **OMM**
- **BOR**
- **QuizRDF**
- **Spectacle**