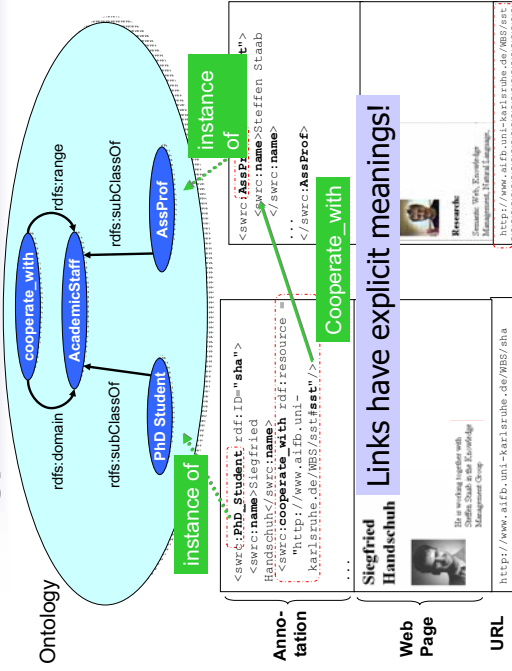


## Ontology & Metadata



3

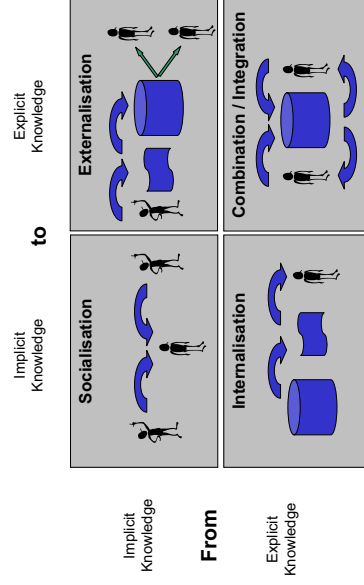
# Intelligent Systems on the World Wide Web

## Ontology Lifecycle



1

## Explicit vs. Implicit Knowledge



Slide 4

4

## 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

Slide 2

2

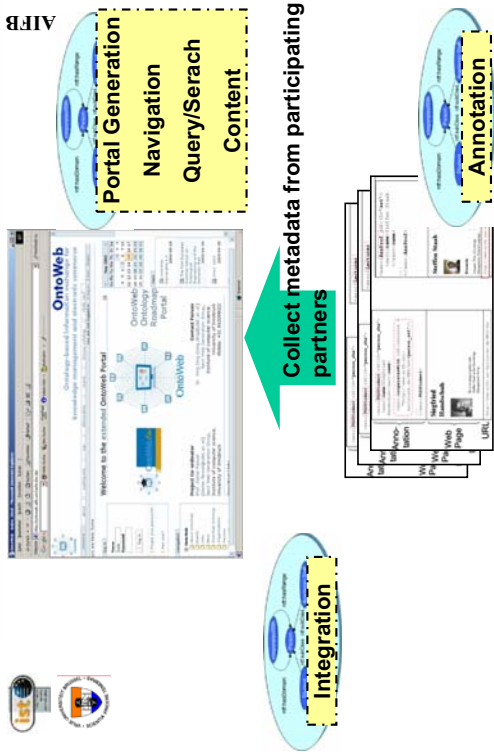
■ **Task:** Build ontology based KM applications

- **Problems:**
  - **Collaboration** between domain experts and knowledge engineers
  - **Evaluation** of ontologies

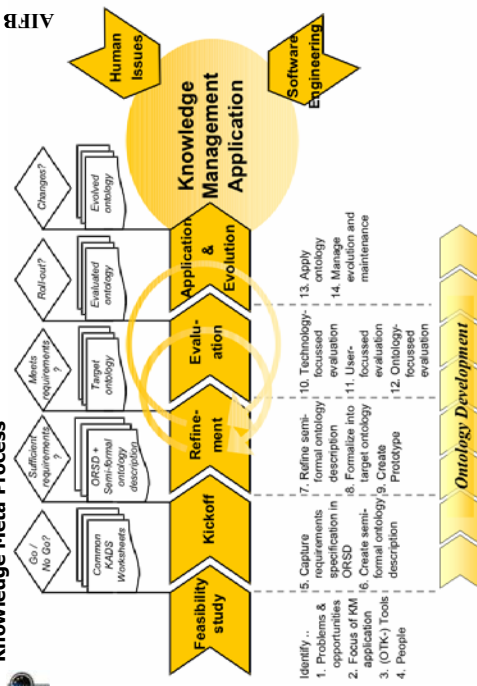
■ Process-oriented, cyclic  
 ■ Pre-defined decisions and outcomes for each step  
 ■ Links to further existing methodologies for substeps

Slide 7

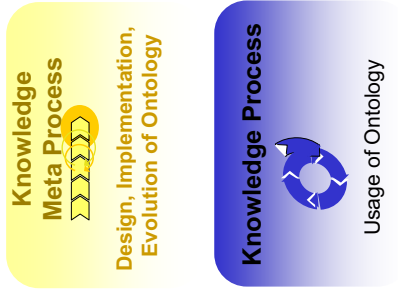
**Case study: OntoWeb.org**



**OTK Methodology: Knowledge Meta Process**



**Ontology-based Processes**



Slide 6

## Feasibility Study

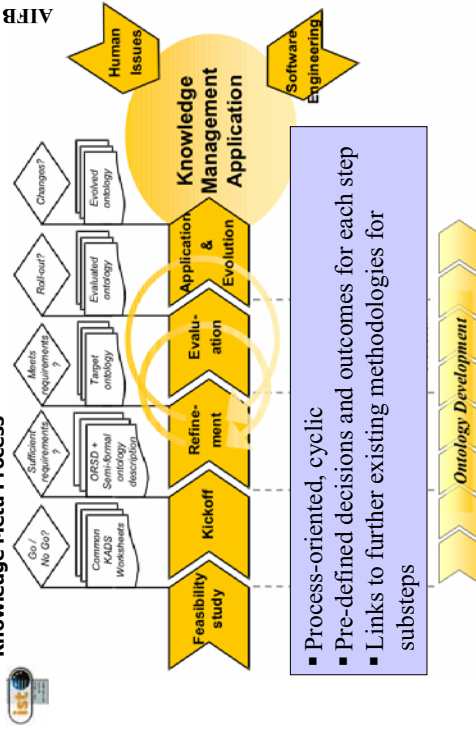
- KM systems only function satisfactorily if they are properly integrated into the organization
- Many factors other than technology determine the success of such a system
- (Based on CommonKADS)

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

Slide 11

11

## OTK Methodology: Knowledge Meta Process



9

## Feasibility study Current State: Skills Management

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

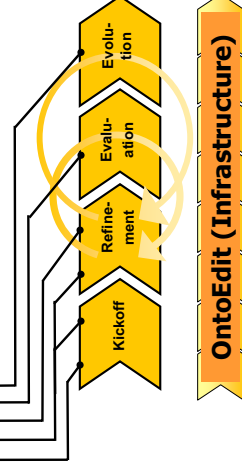


Slide 12

12

## Tools

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

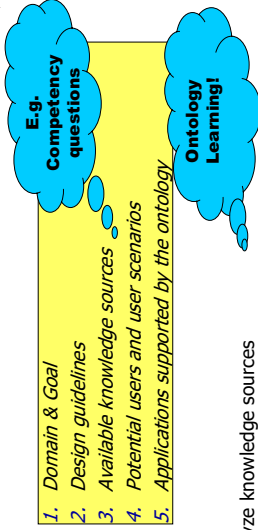


Slide 10

10

## Ontology Kickoff

- Ontology Requirements Specification Document (ORSD)

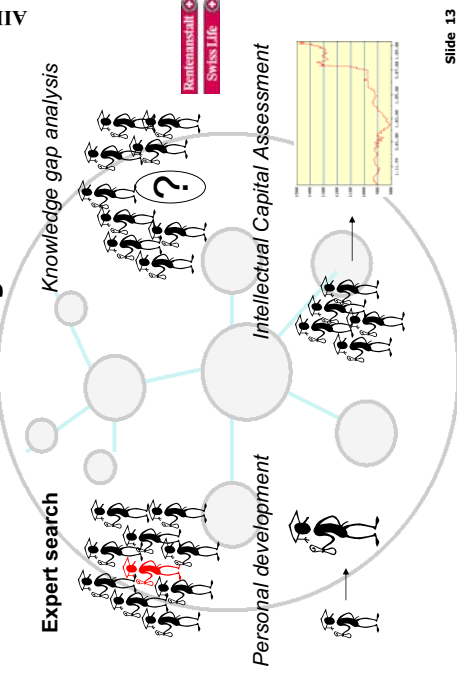


- Analyze knowledge sources
- Develop **baseline ontology description**

*Draft version*, typically most important concepts and relations are identified and described as an untyped graph

Slide 15

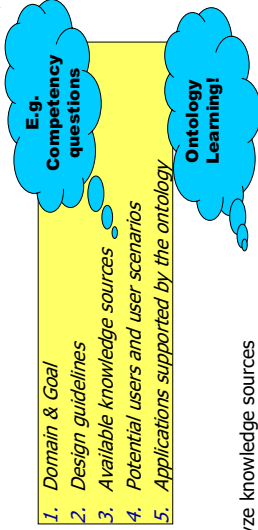
## Feasibility study Intended state: Skills Management



Slide 13

## Ontology Kickoff

- Ontology Requirements Specification Document (ORSD)



- Analyze knowledge sources
- Develop **baseline ontology description**

*Draft version*, typically most important concepts and relations are identified and described as an untyped graph

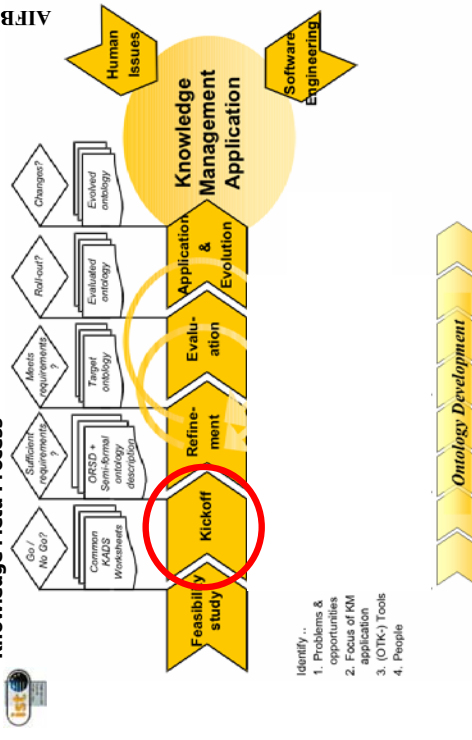
Slide 15

## ORS – Ontology Requirements Specification

- Goal of the ontology:**
  - Tracking and analyzing corporate business histories
- Domain and Scope:**
  - Merger & acquisition, restructurings, 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](http://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

Slide 16

## OTK Methodology: Knowledge Meta Process



## Requirement specification



## CQ – Competency Questionnaire

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

## Requirement specification



## 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

**Kick-off**

# Requirement specification

Design Instructions

- Write all concepts with capital letters.
- Write all relations with small letters.
- Use an underscore (e.g. "ProcessPlant").
- If you have only one concept as a sub-concept, embed it in a sub-heading!

Estimated number of concepts: 500

Maximal depth of concept hierarchy: 4

< Back Forward >

**Kick-off**

# Knowledge Sources

Source	Type	Status
CO C Tool Source - E... 11... 11...	Competency Questionnaire	NEW SOURCE
university	Ontology	NEW SOURCE
word	Word Document	NEW SOURCE
html	HTML Document	NEW SOURCE
xml	XML Document	NEW SOURCE
text	Text Document	NEW SOURCE

1 2 3

**Kick-off**

# Knowledge Sources

1 2

**Kick-off**

# Knowledge Sources

1 3



# Competency questions

Slide 27



# Competency questions

Slide 25



# Competency questions

Slide 28



# Competency questions

Slide 26



**Kick-off**

**Brainstorming, Structuring, Formalisation**

York Sun, 2003

AIRB

33

**Kick-off**

**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

York Sun, 2003

AIRB

Slide 34

**Kick-off**

**Mind2Onto**

York Sun, 2003

AIRB

Slide 35

**Kick-off**

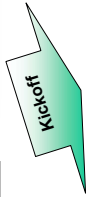
**Mind2Onto**

York Sun, 2003

AIRB

Slide 36

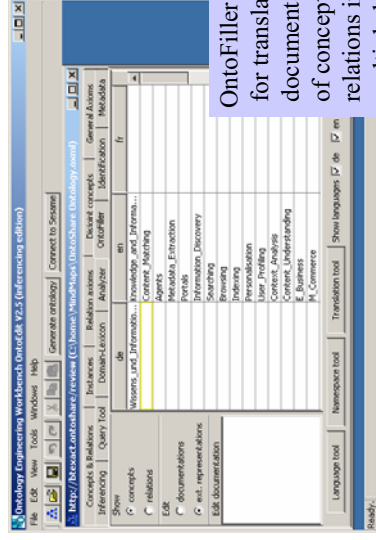
# Refinement



- 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*

Slide 39

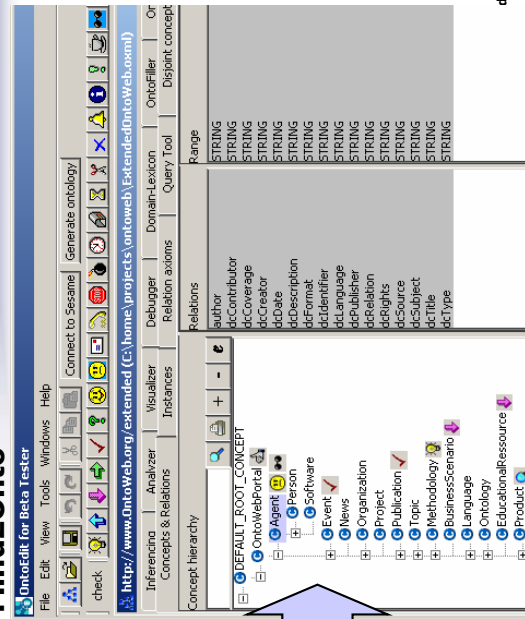
# OntoEdit/OntoFiller



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

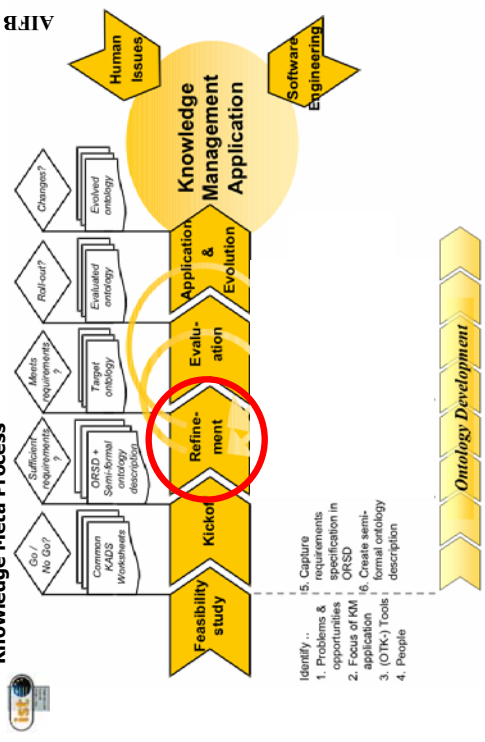
Slide 37

# Mind2Onto



de 40

# OTK Methodology: Knowledge Meta Process



**Refinement**

**Exploit Inferencing**

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

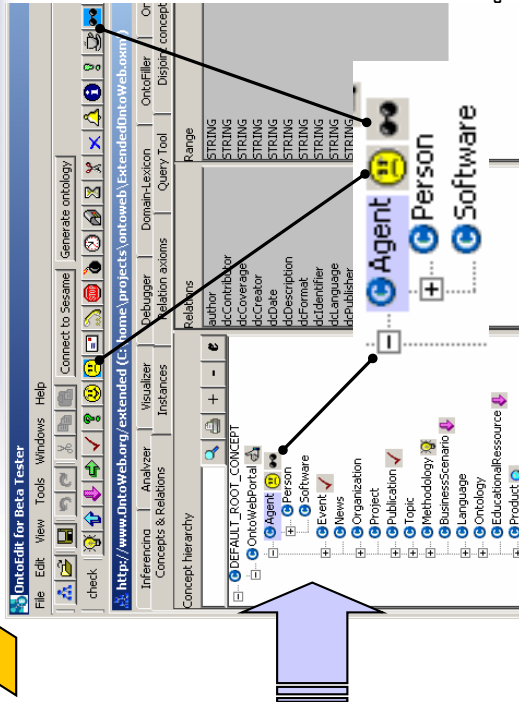


Contrast: OIIEd

Slide 43

**Refinement**

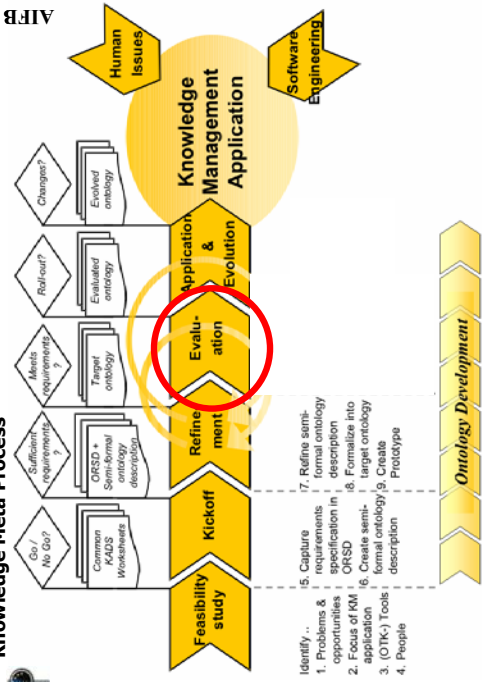
**Mind2Onto**



de 41

**Refinement**

**OTK Methodology: Knowledge Meta Process**



**Refinement**

**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))

Slide 42

## 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.

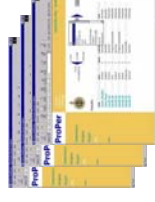
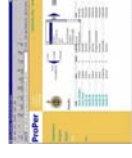
**Example:** „Person” is necessarily an essential property for all its individuals.

- There exist similar definitions for „identity” (+I, -I, +O, -O), „unity” (+U, -U, ~U), „dependency” (+D, -D), ... etc. ...

47

## Evaluation

- Check requirements (ORSD)
  - Are all CQs answered?
  - Is the ontology within the scope?
- Test in target application
  - Analyze usage patterns
- Deploy application(s)



Slide 45

45

## OntoClean: Classification & ideal structure

	+D	+R	-D	-R	Type
					Quasi-type
					Material role
					Phased sortal
					Mixin
					Category
					Formal Role
					Attribution
					Non-sortal
					incoherent
+O	+I	-R	-R	-R	

See: [Welty & Guarino, 2001]

Slide 48

48

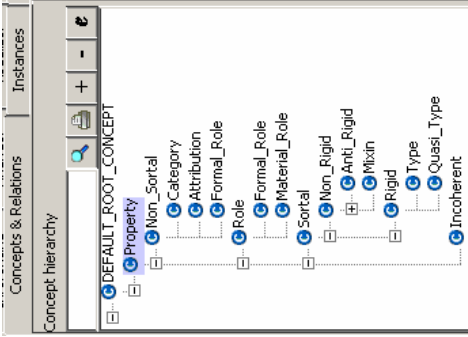
## OntoClean

- **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

Slide 46

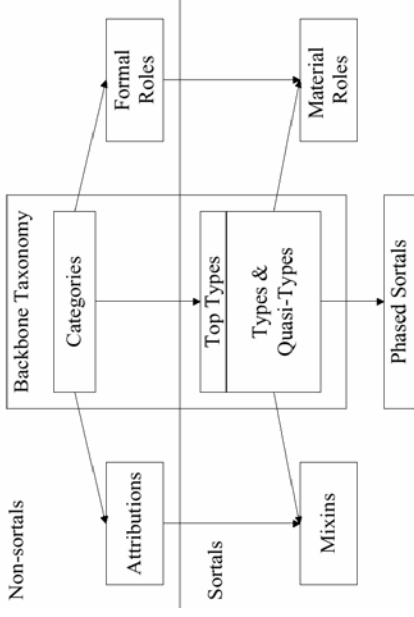
46

## OntoCleanPlugin: Formalisation of meta ontology



Slide 51

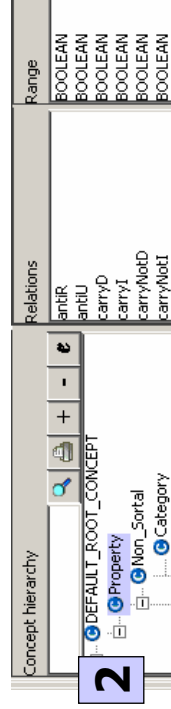
## OntoClean: Classification & ideal structure



See: [Weity & Guarino, 2001.]

Slide 49

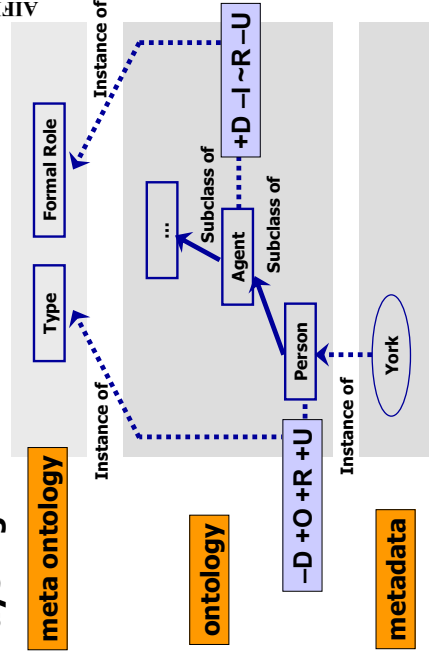
## OntoCleanPlugin: Formalisation of meta ontology



Uppermost concept „Property“ of the *meta ontology* has attached all relations necessary for classifying concepts of an *ontology*

Slide 52

## OntoClean: Layering



## OntoCleanPlugin: Cleaning example

**+D -I ~R -U**  
Def.: Being an active participant in some event.

The screenshot shows the OntoCleanPlugin interface with the following elements:

- Class Hierarchy:**
  - Agent** (highlighted in yellow)
  - Person**
  - D +O +R +U** (highlighted in blue)
- Table:** A table with columns for 'Class' and 'Instance'. The 'Agent' class has a checkmark in the 'Instance' column. The 'Person' class has a checkmark in the 'Instance' column. The '-D +O +R +U' class has a checkmark in the 'Instance' column.

Slide 55

## OntoCleanPlugin: Formalisation of axioms

The screenshot shows the OntoCleanPlugin interface with the following elements:

- Logic Axioms:**

```

FORALL B ( check ("Error: "B," (-R) can't subsume "C," (+R) !")
  <- ( EXISTS C ((C:B
    and (B[#antiR->"true"]
    and C[#carryR->"true"]))) ).
        
```
- Instances:** A table with columns for 'Class' and 'Instance'. The 'Agent' class has a checkmark in the 'Instance' column. The 'Person' class has a checkmark in the 'Instance' column. The '-D +O +R +U' class has a checkmark in the 'Instance' column.

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

## OntoCleanPlugin: Cleaning example

The screenshot shows the OntoCleanPlugin interface with the following elements:

- Logic Axioms:**

```

FORALL V,W,X,Y,Z <- check(V,W,X,Y,Z).
        
```
- Text:**

Evaluating the query FORALL V,W,X,Y,Z <- check(V,W,X,Y,Z).  
Error: Agent (~R) can't subsume Person (+R) !
- Image:** A small image of a man in a suit, likely James, is shown in the bottom right corner.

Slide 56

„Is James an agent?“

## OntoCleanPlugin: Cleaning example

The screenshot shows the OntoCleanPlugin interface with a large table of instances. The table has columns for 'Class' and 'Instance'. The 'Agent' class has a checkmark in the 'Instance' column. The 'Person' class has a checkmark in the 'Instance' column. The '-D +O +R +U' class has a checkmark in the 'Instance' column.

Slide 54

## 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?

Slide 59

59

## OntoCleanPlugin: Cleaning example

The screenshot shows the OntoCleanPlugin interface with the following components:

- Navigation tabs:** Concepts & Relations, Inferencing, Analyzer, Instances, Visualizer, Relation axioms, Debugger, Query Tool, Domain-Lexicon, OntoFiller, Disjoint concepts, OntoClic
- Axioms:**
  - inverse
  - symmetric
  - transitive
  - rigidity\_subsumption
  - unity\_subsumption
  - definition\_4\_5
  - definition\_6\_7
  - definition\_4\_5
  - check rigidity
- Query:** `FORALL V,W,X,Y,Z <- check(V,W,X,Y,Z).`
- Result:** Evaluating the query FORALL V,W,X,Y,Z <- check(V,W,X,Y,Z). Error: Agent (-R) can't subsume person (+R).

Person should not be a subconcept of Agent!

Interpretation: Persons *can be* agents, but persons are not necessarily agents.



„Is James an agent?“

Slide 57

57

## Worksheet for life cycle aspects of metadata

- ala ontology
- Co-ordinated change of data and metadata?
- Co-ordinated change of ontology and metadata?
- Cold start (chicken-and-egg) problem: A problem? How to overcome?
- Granularity of metadata envisaged: classification, people/events/relationships/etc.

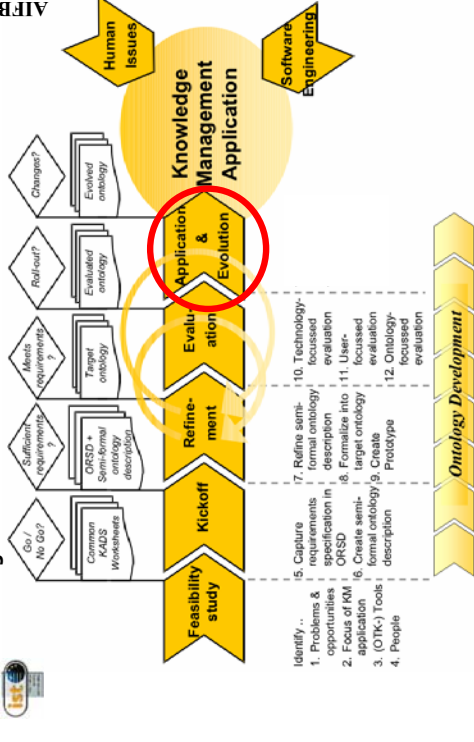
Faustregel – Kosten:

- Hardware 1
- Software 10
- Daten 100

Slide 60

60

## OTK Methodology: Knowledge Meta Process



58

**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"

Slide 63

63

**Coordination of metadata & ontology**

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

Slide 61

61

**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>

Slide 64

64

**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

Slide 62

62

## Requirements for ontology evolution

- Basic requirement
  - **Functional requirement:**
    - enable the handling of the required changes
    - ensure the consistency of the underlying ontology and all dependent artifacts
- Extended requirements
  - **Interaction requirement** – supports the user to manage changes more easily
  - **Refinement requirement** – offers advice to the user for continual system refinement

Slide 67

67

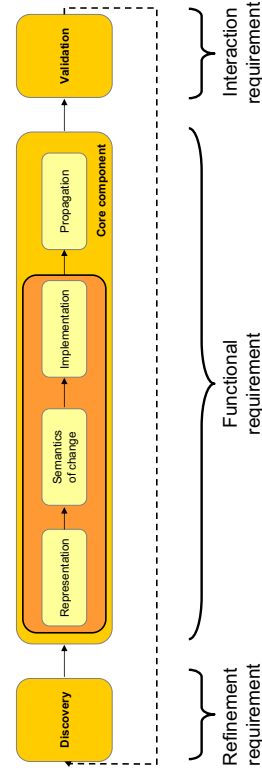
65

## 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. (QMIN) 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

Slide 65

## Ontology Evolution Process



Slide 68

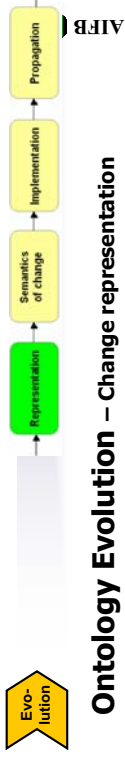
68

## 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**

Slide 66

66

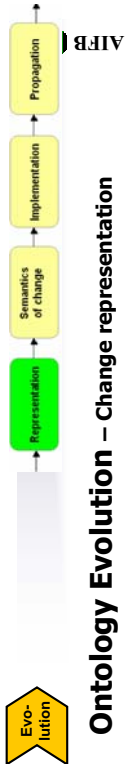


### Ontology Evolution – Change representation

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

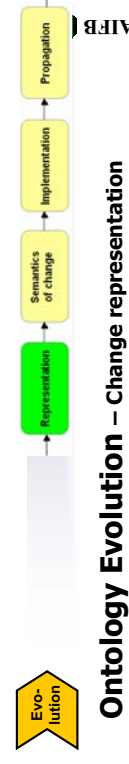
MoveConcept ≠ (RemoveSubConcept + AddSubConcept)

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



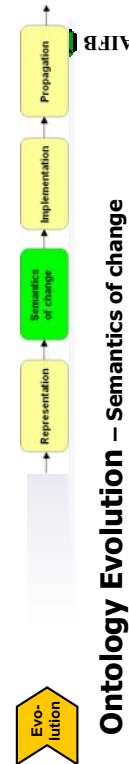
### Ontology Evolution – Change representation

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



### Ontology Evolution – Change representation

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



### Ontology Evolution – Semantics of change

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



## Evolution Strategies

### Elementary evolution strategies

#### Resolution points:

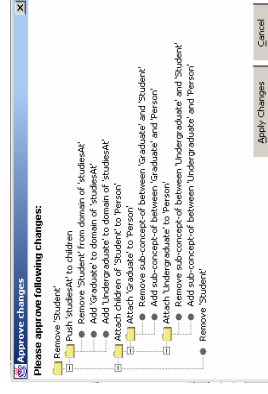
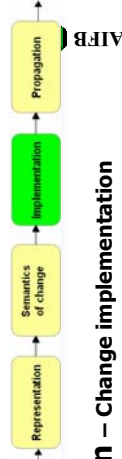
- how to handle orphaned concepts;
- how to handle orphaned properties;
- how to propagate properties to the concept whose parent changes;
- what constitutes a valid domain of a property;
- what constitutes a valid range of a property;
- whether a domain (range) of a property can contain a concept that 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;
- ...

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

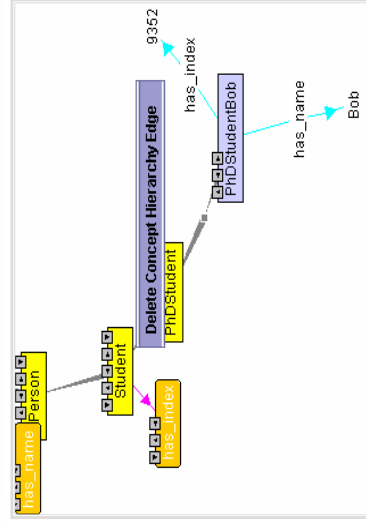
Common policy consisting of a set of elementary evolution strategies, each giving an answer for one resolution point, is an **evolution strategy**

## Ontology Evolution – Change implementation

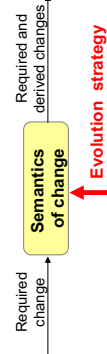
- After user's approval all changes are applied to the ontology
- Since it is necessary to perform several changes together, the **transaction server** is needed.



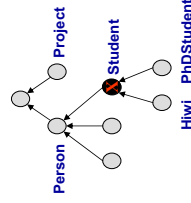
## Example



## Evolution Strategies



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



# Implementation

Applications & Services

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

Middleware

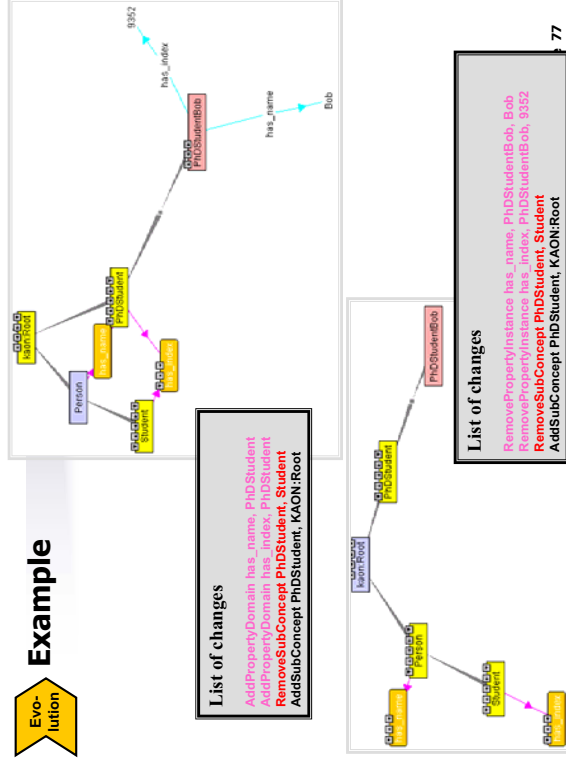
KAON Access Interface			
Evolution Strategy	Reversibility Services	Change Discovery	Interaction Logging
KAON API		Evolution Logging	
RDF API		KAON RDF Server	

Data and Remote Services

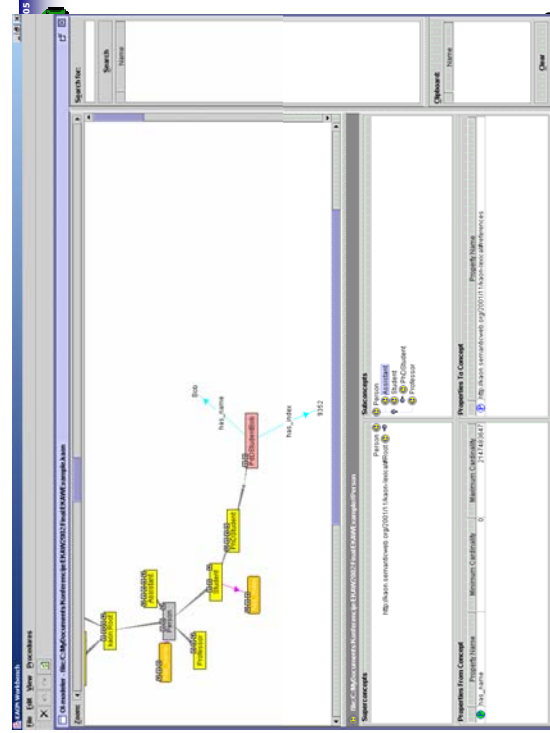
Persistence, Transactions, Security

Slide 79

# Example



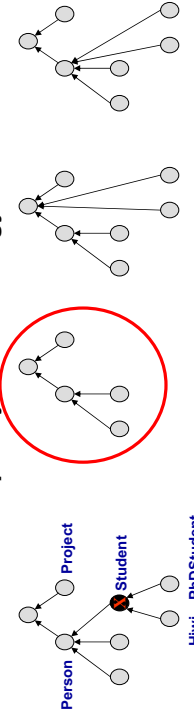
77



# 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*



Slide 78

**Resolution points**

**Elementary evolution strategies**

Orphaned concepts will be...

- ...deleted.
- ...reconnected to ontology root.
- ...reconnected to superconcepts.

When concept's parent is removed...

- ...all inherited properties will be propagated.
- ...all inherited properties will be added to the concept.
- ...only parent's properties will be added to the concept.

Properties without any domain concepts...

- ...may exist in the O-model.
- ...should be deleted from the O-model.

Instance consistency...

- ...should be enforced.
- ...should not be enforced.

When creating a hierarchy path which already exists...

- ...nothing special should be done.
- ...the starter path should be removed.
- ...an error should be raised.

**Resolution points**

**Elementary evolution strategies**

Orphaned concepts will be...

- ...deleted.
- ...reconnected to ontology root.
- ...reconnected to superconcepts.

When concept's parent is removed...

- ...all inherited properties will be propagated.
- ...all inherited properties will be added to the concept.
- ...only parent's properties will be added to the concept.

Properties without any domain concepts...

- ...may exist in the O-model.
- ...should be deleted from the O-model.

Instance consistency...

- ...should be enforced.
- ...should not be enforced.

When creating a hierarchy path which already exists...

- ...nothing special should be done.
- ...the starter path should be removed.
- ...an error should be raised.

**Resolution points**

**Elementary evolution strategies**

Orphaned concepts will be...

- ...deleted.
- ...reconnected to ontology root.
- ...reconnected to superconcepts.

When concept's parent is removed...

- ...all inherited properties will be propagated.
- ...all inherited properties will be added to the concept.
- ...only parent's properties will be added to the concept.

Properties without any domain concepts...

- ...may exist in the O-model.
- ...should be deleted from the O-model.

Instance consistency...

- ...should be enforced.
- ...should not be enforced.

When creating a hierarchy path which already exists...

- ...nothing special should be done.
- ...the starter path should be removed.
- ...an error should be raised.

**Resolution points**

**Elementary evolution strategies**

Orphaned concepts will be...

- ...deleted.
- ...reconnected to ontology root.
- ...reconnected to superconcepts.

When concept's parent is removed...

- ...all inherited properties will be propagated.
- ...all inherited properties will be added to the concept.
- ...only parent's properties will be added to the concept.

Properties without any domain concepts...

- ...may exist in the O-model.
- ...should be deleted from the O-model.

Instance consistency...

- ...should be enforced.
- ...should not be enforced.

When creating a hierarchy path which already exists...

- ...nothing special should be done.
- ...the starter path should be removed.
- ...an error should be raised.

## Conclusions on Knowledge Meta Process

87



### 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

Slide 85

85

## Experiences from OTK Case Studies

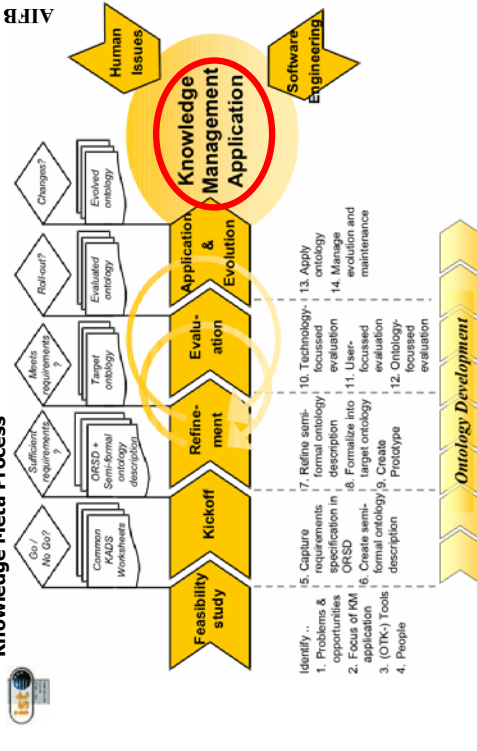
- **Guidelines** for domain experts from industry have to be pragmatic
  1. Train the user about ontologies
  2. Show the concrete advantage of the KMS
  3. Model precisely – but allow for imprecise views (most users cannot distinguish classes vs instances or isa vs partOf)
- **Plan for Maintenance**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

Slide 88

88

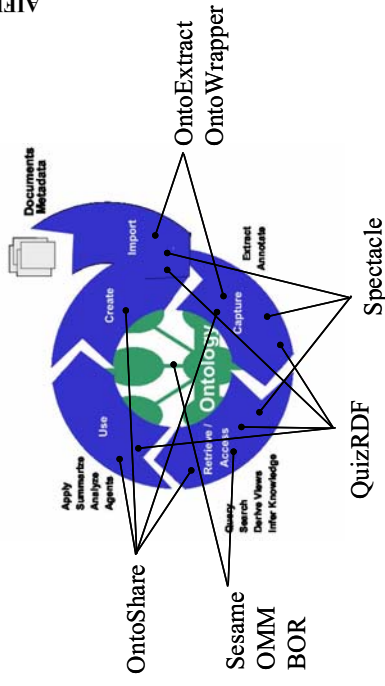


### OTK Methodology: Knowledge Meta Process



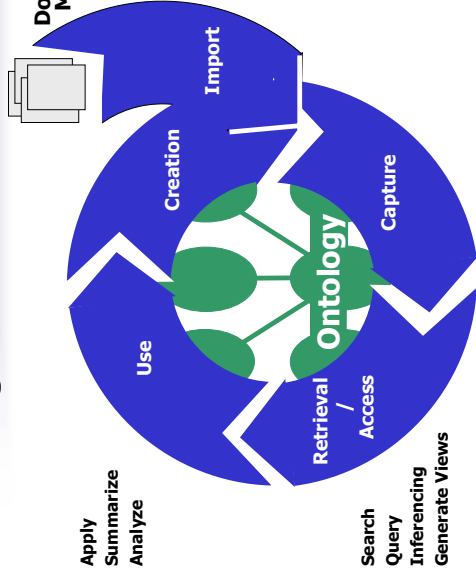
86

# Users Portal



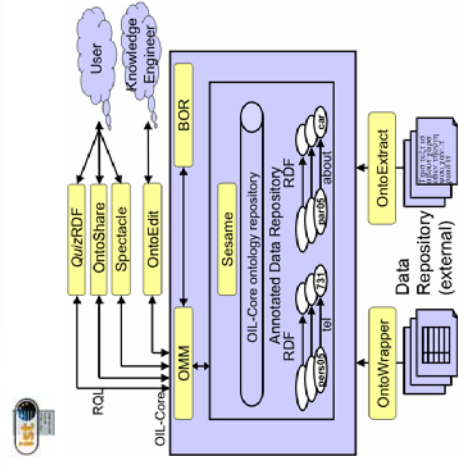
Slide 91

# Knowledge Process



Slide 89

# OTK Architecture



Slide 92

# OTK Case Study @ BT

