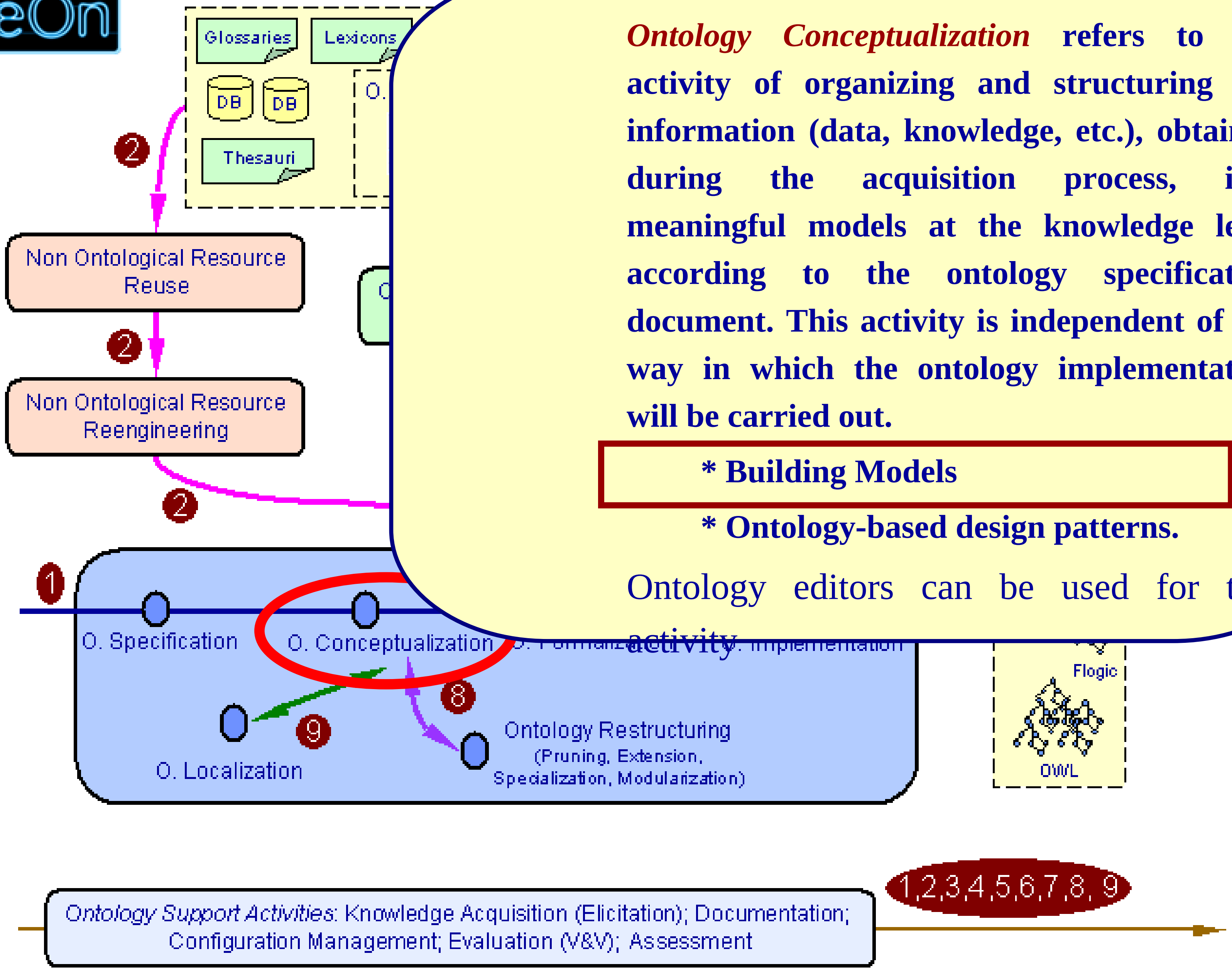

Glossaries
Lexicons
DB
DB
Thesauri
O.
2
Non Ontological Resource Reuse
2
Non Ontological Resource Reengineering
2
1
O. Specification
O. Conceptualization
O. Formaliza
O. Implementation
9
8
O. Localization
Ontology Restructuring
(Pruning, Extension, Specialization, Modularization)
Flogic
OWL
1,2,3,4,5,6,7,8, 9
Ontology Support Activities: Knowledge Acquisition (Elicitation); Documentation; Configuration Management; Evaluation (V&V); Assessment

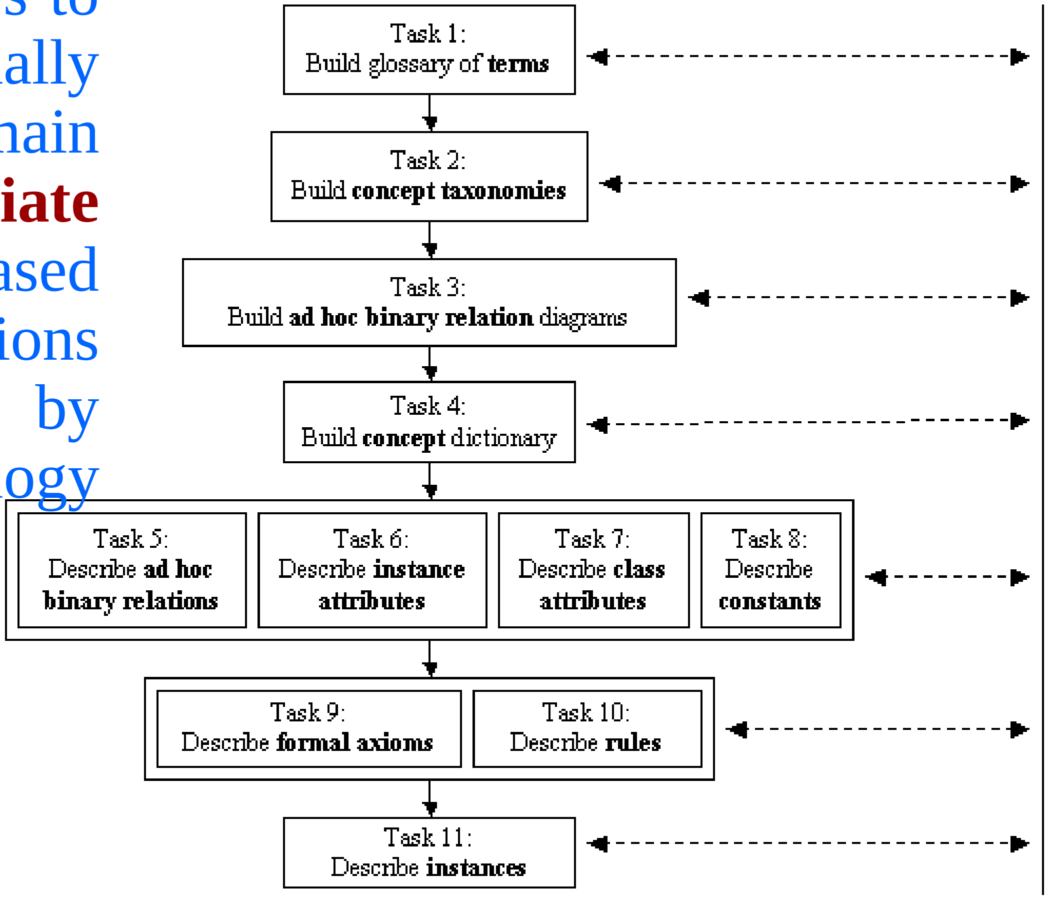
Ontology Conceptualization **refers to the activity of organizing and structuring the information (data, knowledge, etc.), obtained during the acquisition process, into meaningful models at the knowledge level according to the ontology specification document. This activity is independent of the way in which the ontology implementation will be carried out.**

- **Building Models**
- **Ontology-based design patterns.**

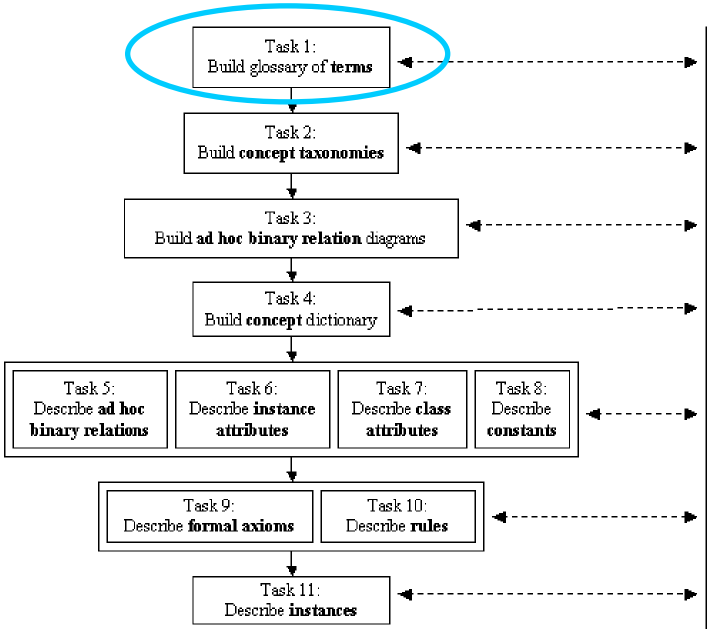
Ontology editors can be used for this activity.

METHONTOLOGY. Tasks in the conceptualization activity

METHONTOLOGY proposes to organize the informally perceived view of a domain using a set of **intermediate representations (IRs)** based on tabular and graph notations that can be understood by domain experts and ontology developers.


Task 1:
Build glossary of **terms**
Task 2:
Build **concept taxonomies**
Task 3:
Build **ad hoc binary relation** diagrams
Task 4:
Build **concept** dictionary
Task 5:
Describe **ad hoc binary relations**
Task 6:
Describe **instance attributes**
Task 7:
Describe **class attributes**
Task 8:
Describe **constants**
Task 9:
Describe **formal axioms**
Task 10:
Describe **rules**
Task 11:
Describe **instances**

METHONTOLOGY. Terms Glossary



Task 1: Build glossary of **terms**

Task 2: Build **concept taxonomies**

Task 3: Build **ad hoc binary relation** diagrams

Task 4: Build **concept** dictionary

Task 5: Describe **ad hoc binary relations**

Task 6: Describe **instance attributes**

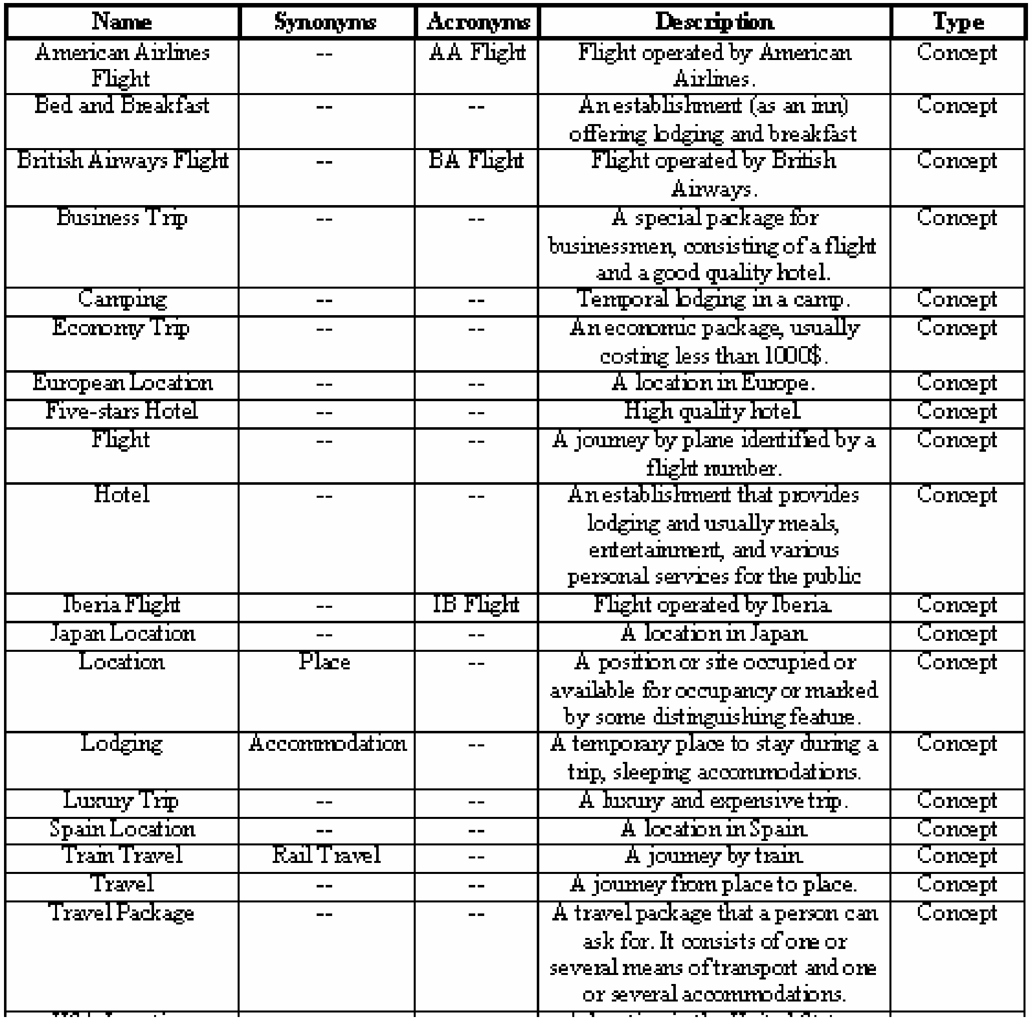
Task 7: Describe **class attributes**

Task 8: Describe **constants**

Task 9: Describe **formal axioms**

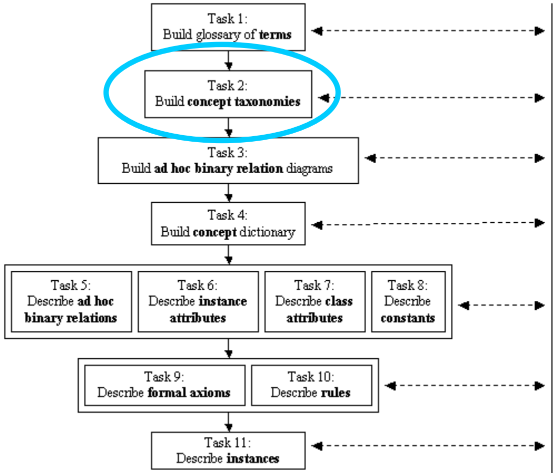
Task 10: Describe **rules**

Task 11: Describe **instances**



Name	Synonyms	Acronyms	Description	Type
American Airlines Flight	--	AA Flight	Flight operated by American Airlines.	Concept
Bed and Breakfast	--	--	An establishment (as an inn) offering lodging and breakfast	Concept
British Airways Flight	--	BA Flight	Flight operated by British Airways.	Concept
Business Trip	--	--	A special package for businessmen, consisting of a flight and a good quality hotel.	Concept
Camping	--	--	Temporal lodging in a camp.	Concept
Economy Trip	--	--	An economic package, usually costing less than 1000$.	Concept
European Location	--	--	A location in Europe.	Concept
Five-stars Hotel	--	--	High quality hotel	Concept
Flight	--	--	A journey by plane identified by a flight number.	Concept
Hotel	--	--	An establishment that provides lodging and usually meals, entertainment, and various personal services for the public	Concept
Iberia Flight	--	IB Flight	Flight operated by Iberia.	Concept
Japan Location	--	--	A location in Japan.	Concept
Location	Place	--	A position or site occupied or available for occupancy or marked by some distinguishing feature.	Concept
Lodging	Accommodation	--	A temporary place to stay during a trip, sleeping accommodations.	Concept
Luxury Trip	--	--	A luxury and expensive trip.	Concept
Spain Location	--	--	A location in Spain.	Concept
Train Travel	Rail Travel	--	A journey by train.	Concept
Travel	--	--	A journey from place to place.	Concept
Travel Package	--	--	A travel package that a person can ask for. It consists of one or several means of transport and one or several accommodations.	Concept

METHONTOLOGY. Primitives for Modelling Taxonomies



Task 1: Build glossary of **terms**

Task 2: Build **concept taxonomies**

Task 3: Build **ad hoc binary relation** diagrams

Task 4: Build **concept** dictionary

Task 5: Describe **ad hoc binary relations**

Task 6: Describe **instance attributes**

Task 7: Describe **class attributes**

Task 8: Describe **constants**

Task 9: Describe **formal axioms**

Task 10: Describe **rules**

Task 11: Describe **instances**

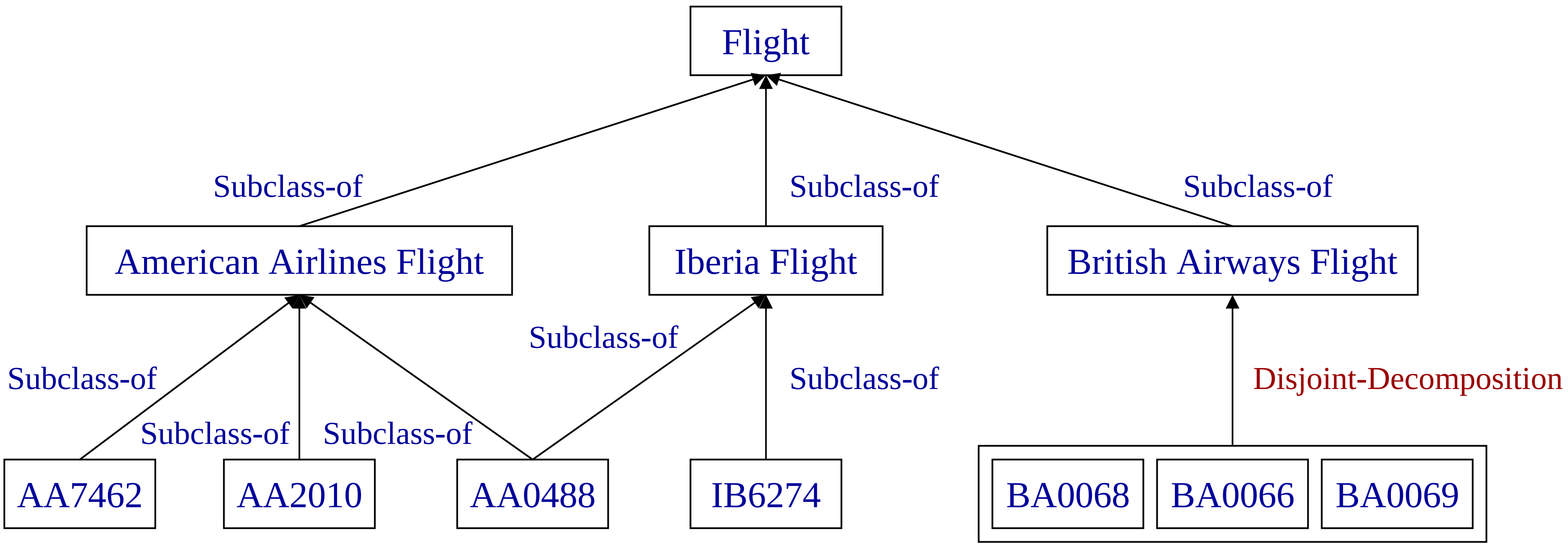
Subclass-of:

Disjoint decomposition: **a set of subclasses of C that do not have common instances and do not cover C**

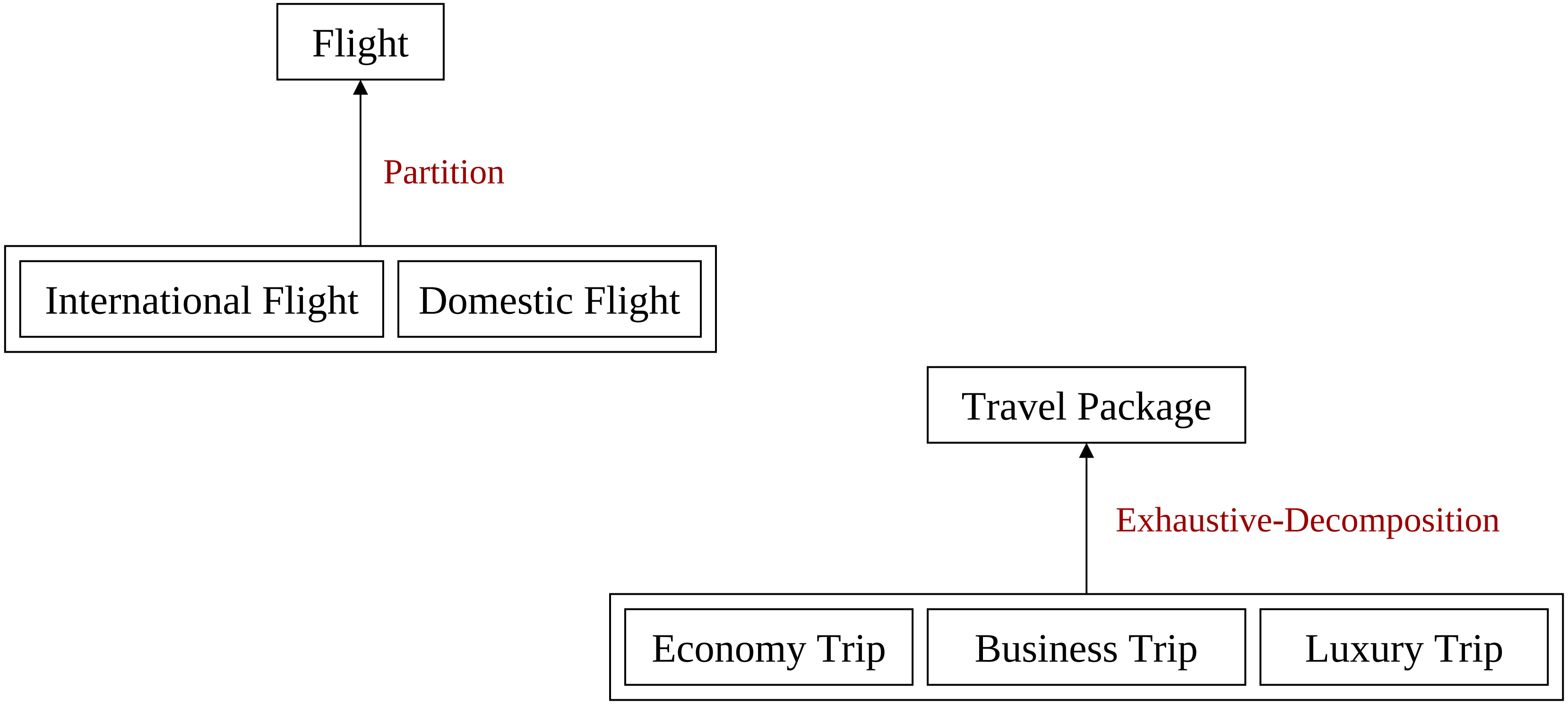
Partition: **a set subclasses of C that cover C and do not have common instances or subclasses**

Exhaustive-Decomposition: **a set subclasses of C that cover C and may have common instances or subclasses**

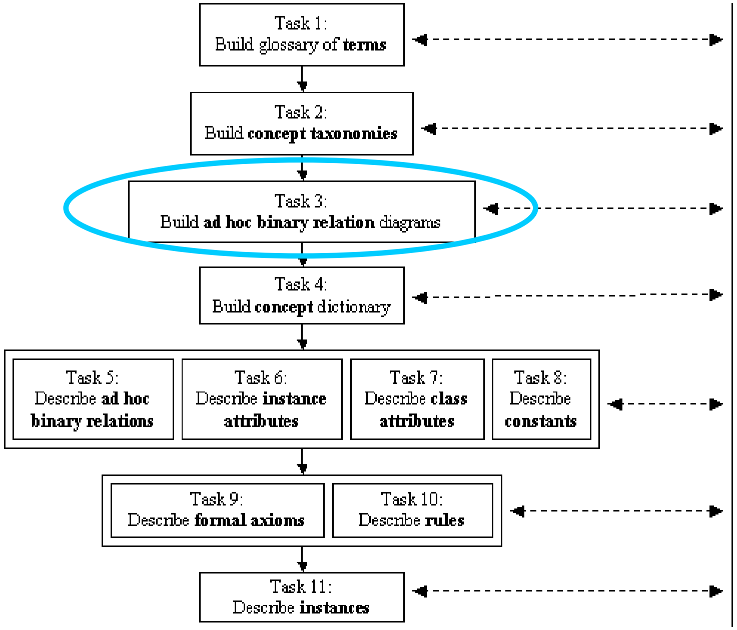
Example of a Taxonomy (I)



Example of a Taxonomy (II)



METHONTOLOGY. Identify Ad-hoc relations



Task 1:
Build glossary of **terms**

Task 2:
Build **concept taxonomies**

Task 3:
Build **ad hoc binary relation** diagrams

Task 4:
Build **concept** dictionary

Task 5:
Describe **ad hoc binary relations**

Task 6:
Describe **instance attributes**

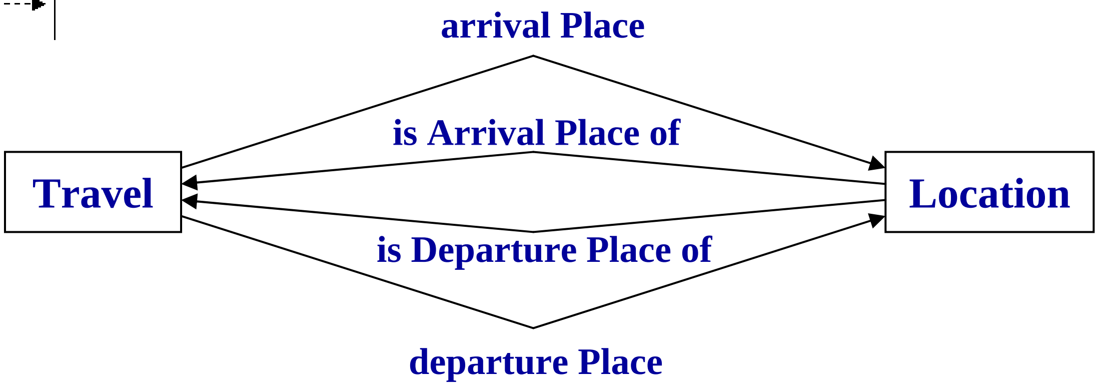
Task 7:
Describe **class attributes**

Task 8:
Describe **constants**

Task 9:
Describe **formal axioms**

Task 10:
Describe **rules**

Task 11:
Describe **instances**



arrival Place

is Arrival Place of

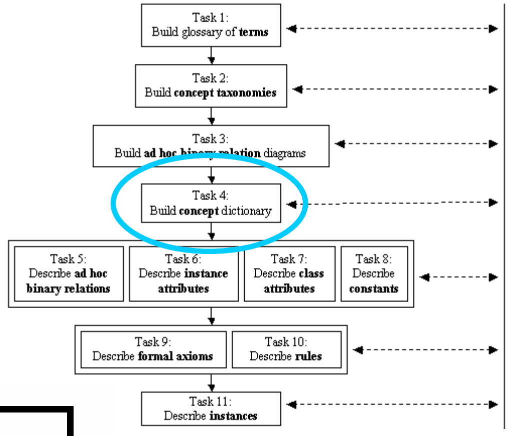
Travel

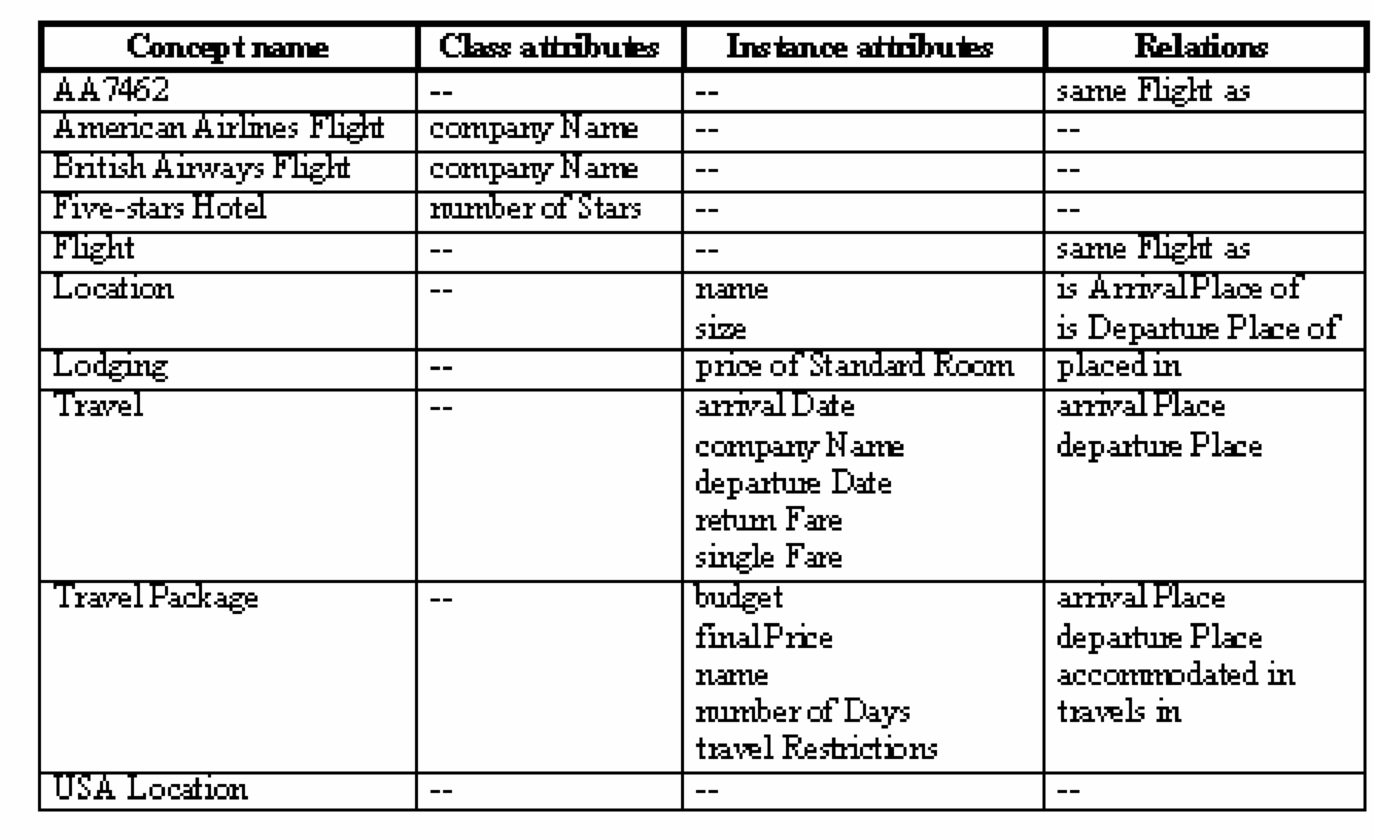
Location

is Departure Place of

departure Place

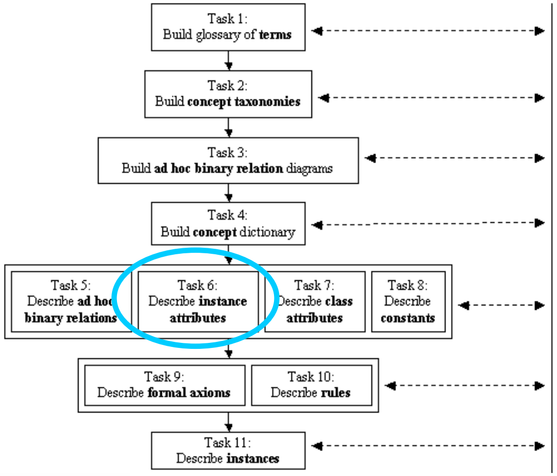
METHONTOLOGY. Define a Concept Dictionary

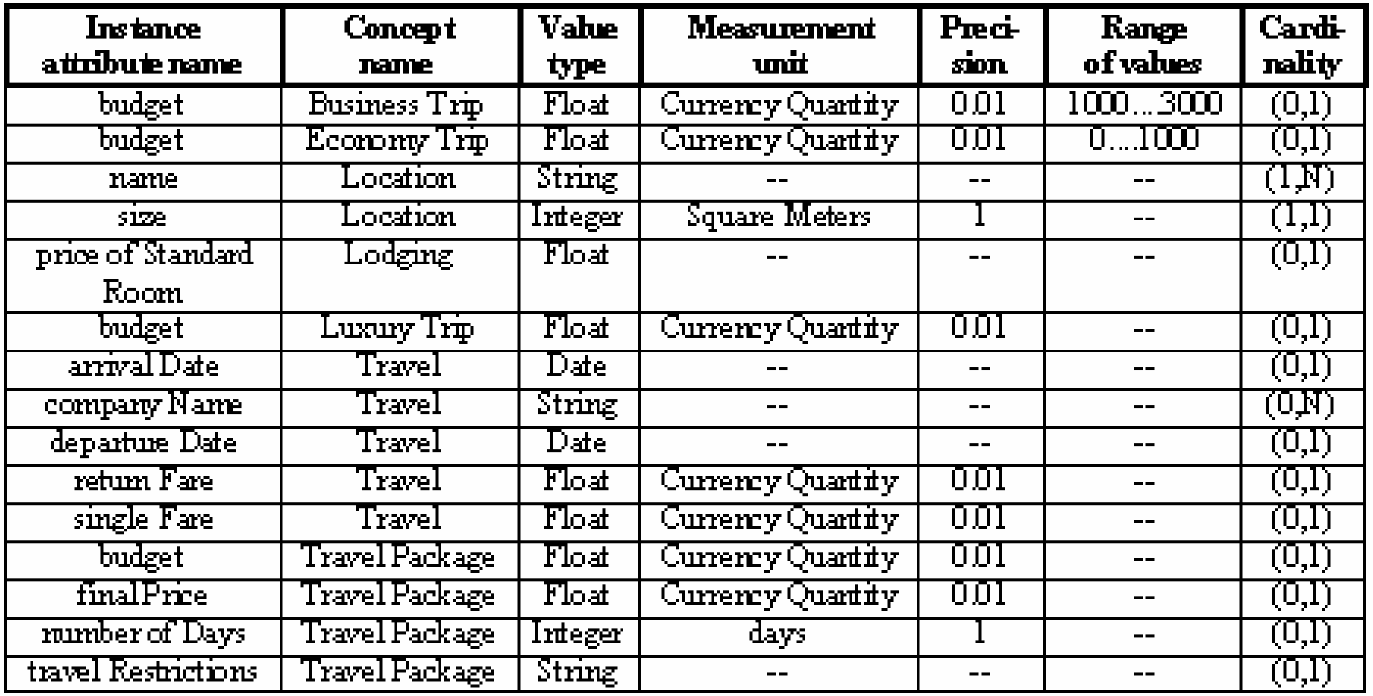




Concept name	Class attributes	Instance attributes	Relations
AA7462	--	--	same Flight as
American Airlines Flight	company Name	--	--
British Airways Flight	company Name	--	--
Five-stars Hotel	number of Stars	--	--
Flight	--	--	same Flight as
Location	--	name size	is Arrival Place of is Departure Place of
Lodging	--	price of Standard Room	placed in
Travel	--	arrival Date company Name departure Date return Fare single Fare	arrival Place departure Place
Travel Package	--	budget final Price name number of Days travel Restrictions	arrival Place departure Place accommodated in travels in
USA Location	--	--	--

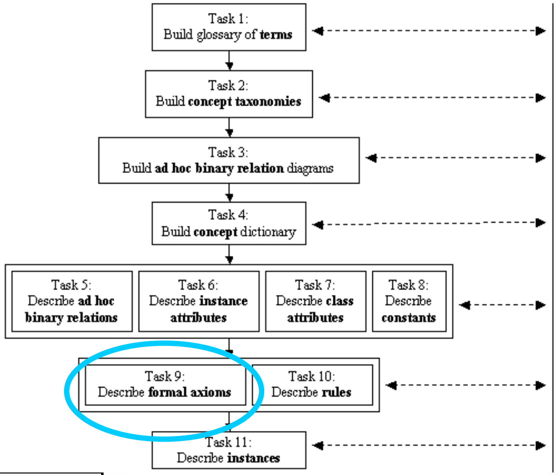
METHONTOLOGY. Define Instance Attributes





Instance attribute name	Concept name	Value type	Measurement unit	Preci-sion	Range of values	Cardi-nality
budget	Business Trip	Float	Currency Quantity	0.01	1000...3000	(0,1)
budget	Economy Trip	Float	Currency Quantity	0.01	0....1000	(0,1)
name	Location	String	--	--	--	(1,N)
size	Location	Integer	Square Meters	1	--	(1,1)
price of Standard Room	Lodging	Float	--	--	--	(0,1)
budget	Luxury Trip	Float	Currency Quantity	0.01	--	(0,1)
arrival Date	Travel	Date	--	--	--	(0,1)
company Name	Travel	String	--	--	--	(0,N)
departure Date	Travel	Date	--	--	--	(0,1)
return Fare	Travel	Float	Currency Quantity	0.01	--	(0,1)
single Fare	Travel	Float	Currency Quantity	0.01	--	(0,1)
budget	Travel Package	Float	Currency Quantity	0.01	--	(0,1)
final Price	Travel Package	Float	Currency Quantity	0.01	--	(0,1)
number of Days	Travel Package	Integer	days	1	--	(0,1)
travel Restrictions	Travel Package	String	--	--	--	(0,1)

METHONTOLOGY. Define Formal Axioms



Task 1: Build glossary of **terms**

Task 2: Build **concept taxonomies**

Task 3: Build **ad hoc binary relation** diagrams

Task 4: Build **concept** dictionary

Task 5: Describe **ad hoc binary relations**

Task 6: Describe **instance attributes**

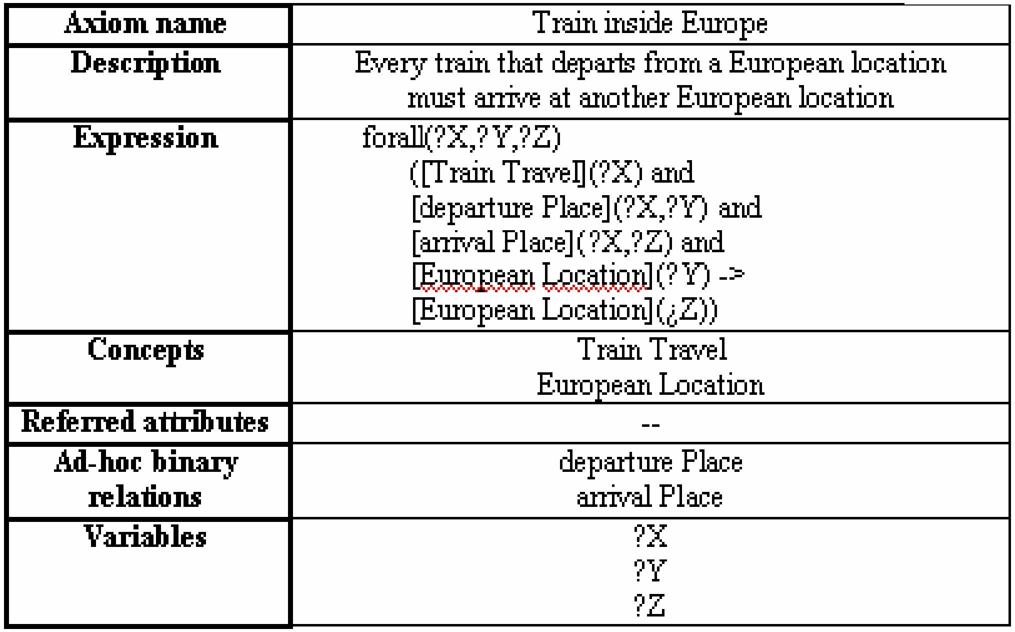
Task 7: Describe **class attributes**

Task 8: Describe **constants**

Task 9: Describe **formal axioms**

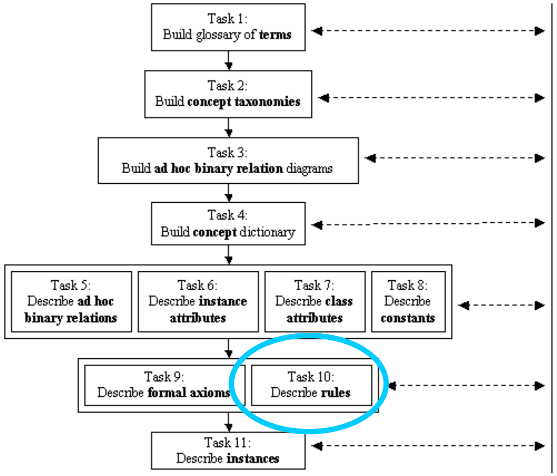
Task 10: Describe **rules**

Task 11: Describe **instances**



Axiom name	Train inside Europe
Description	Every train that departs from a European location must arrive at another European location
Expression	forall(?X,?Y,?Z) ([Train Travel](?X) and [departure Place](?X,?Y) and [arrival Place](?X,?Z) and [European Location](?Y) -> [European Location](¿Z))
Concepts	Train Travel European Location
Referred attributes	--
Ad-hoc binary relations	departure Place arrival Place
Variables	?X ?Y ?Z

METHONTOLOGY. Define Rules


Task 1: Build glossary of **terms**

Task 2: Build **concept taxonomies**

Task 3: Build **ad hoc binary relation** diagrams

Task 4: Build **concept** dictionary

Task 5: Describe **ad hoc binary relations**

Task 6: Describe **instance attributes**

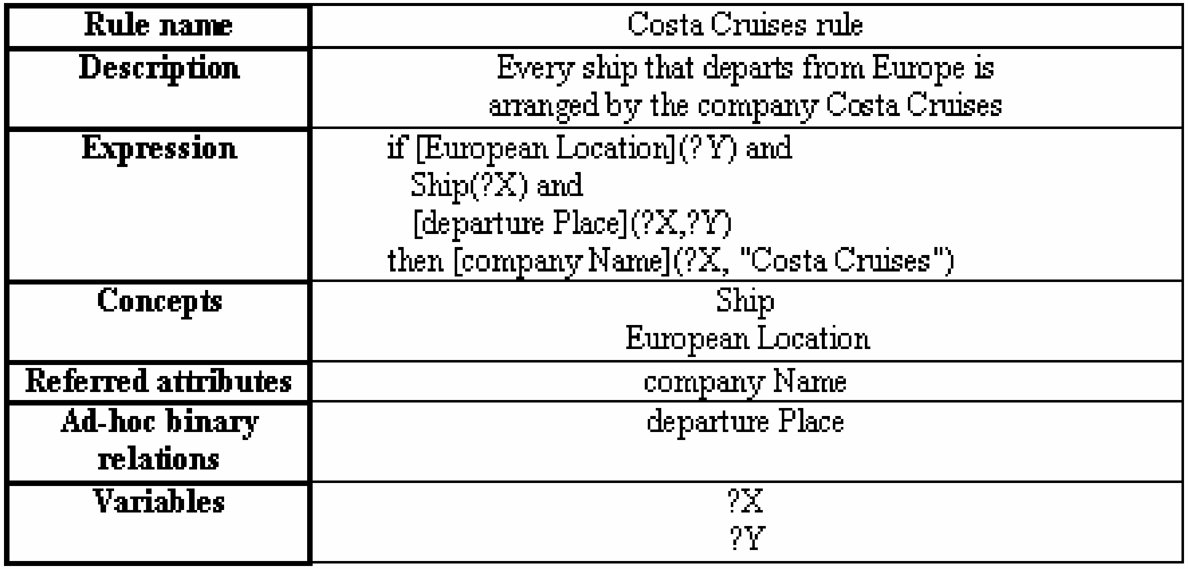
Task 7: Describe **class attributes**

Task 8: Describe **constants**

Task 9: Describe **formal axioms**

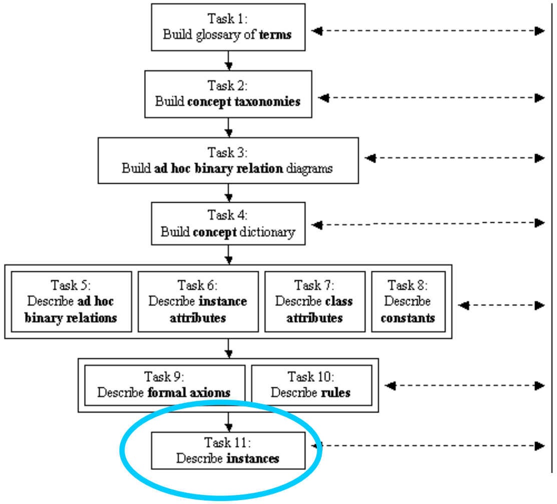
Task 10: Describe **rules**

Task 11: Describe **instances**



Rule name	Costa Cruises rule
Description	Every ship that departs from Europe is arranged by the company Costa Cruises
Expression	if [European Location](?Y) and Ship(?X) and [departure Place](?X,?Y) then [company Name](?X, "Costa Cruises")
Concepts	Ship European Location
Referred attributes	company Name
Ad-hoc binary relations	departure Place
Variables	?X ?Y

METHONTOLOGY. Define Instances


Task 1:
Build glossary of **terms**

Task 2:
Build **concept taxonomies**

Task 3:
Build **ad hoc binary relation** diagrams

Task 4:
Build **concept** dictionary

Task 5:
Describe **ad hoc binary relations**

Task 6:
Describe **instance attributes**

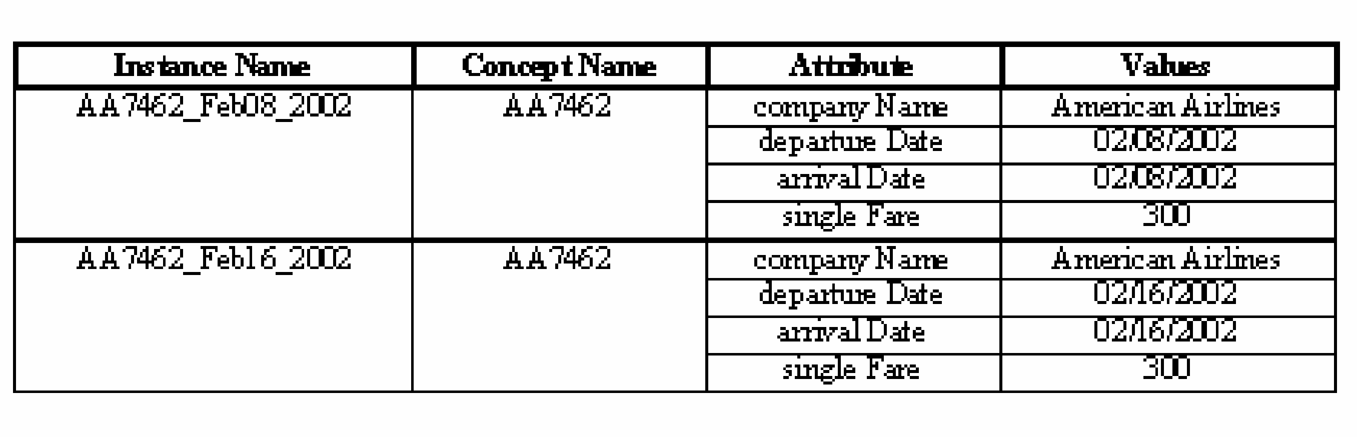
Task 7:
Describe **class attributes**

Task 8:
Describe **constants**

Task 9:
Describe **formal axioms**

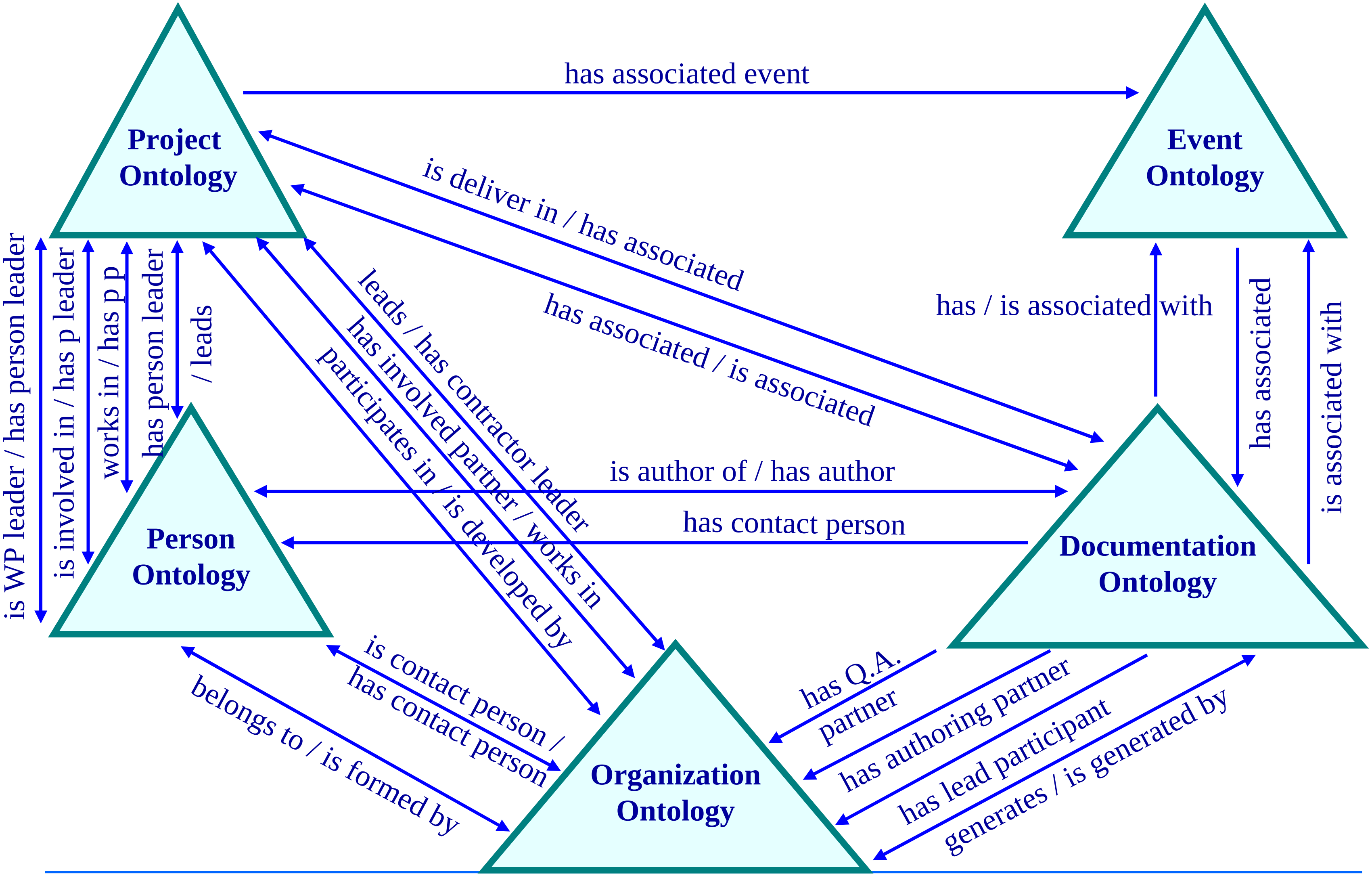
Task 10:
Describe **rules**

Task 11:
Describe **instances**

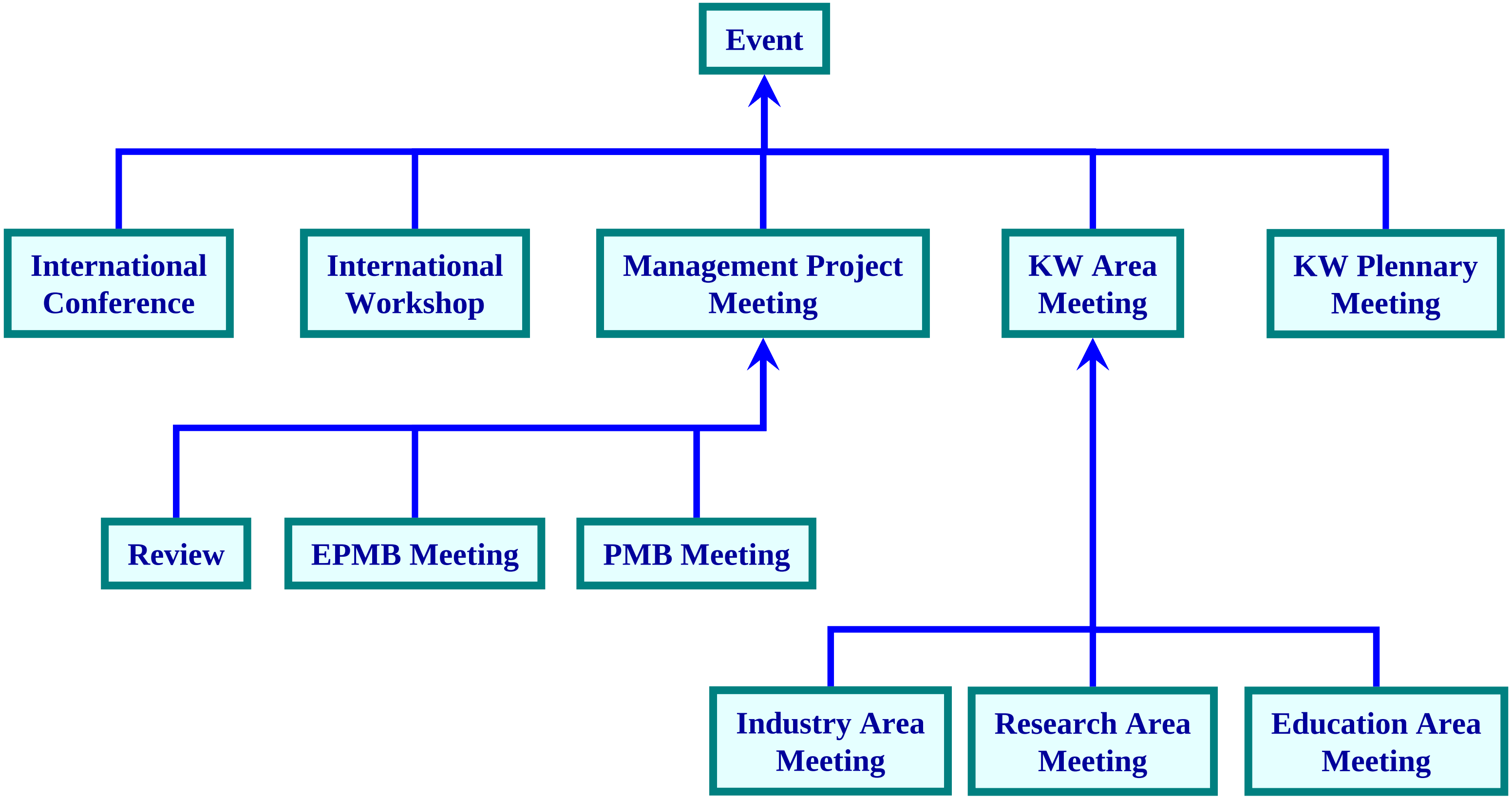


Instance Name	Concept Name	Attribute	Values
AA7462_Feb08_2002	AA7462	company Name	American Airlines
		departure Date	02/08/2002
		arrival Date	02/08/2002
		single Fare	300
AA7462_Feb16_2002	AA7462	company Name	American Airlines
		departure Date	02/16/2002
		arrival Date	02/16/2002
		single Fare	300

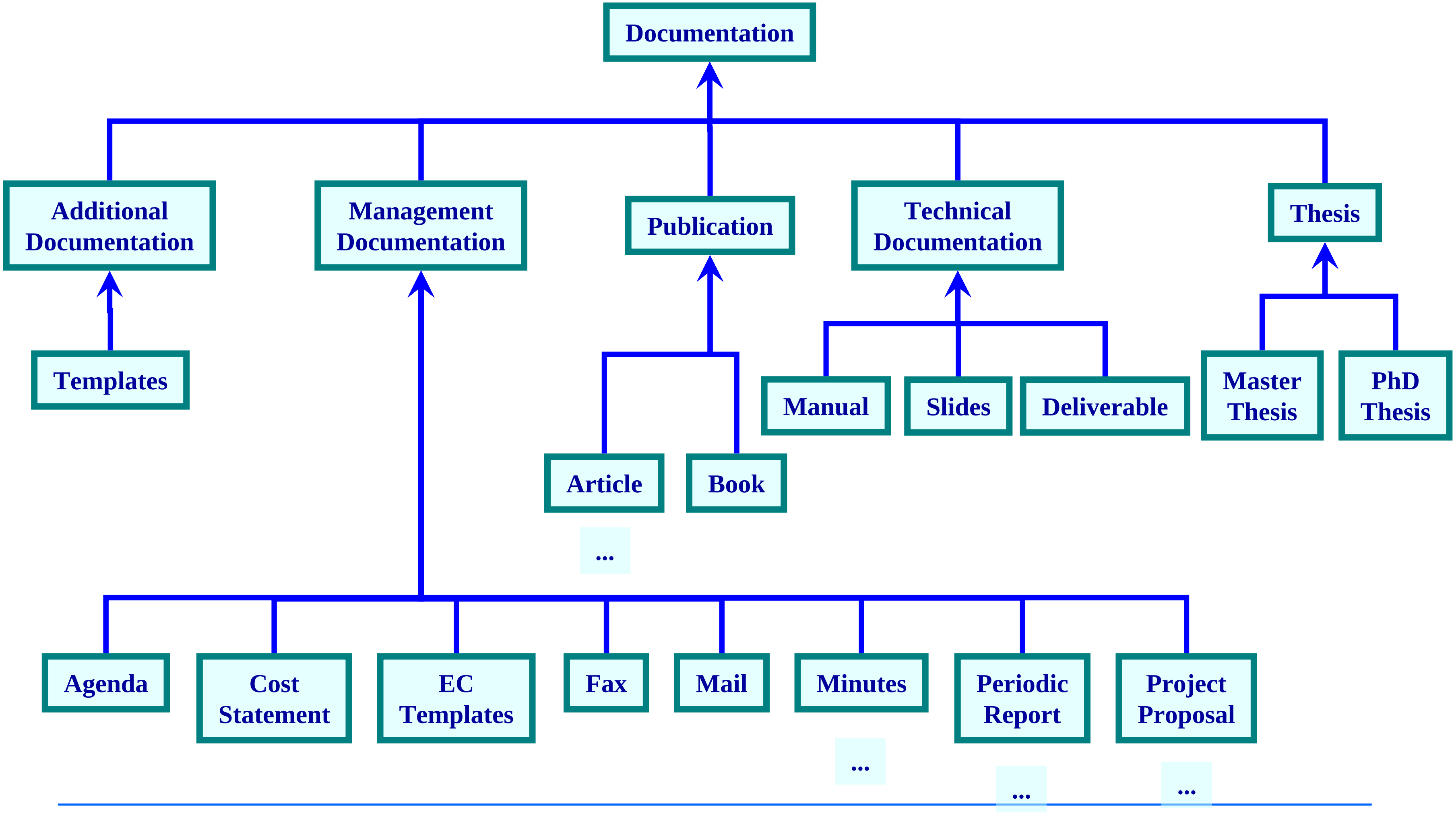
Knowledge Web Ontologies



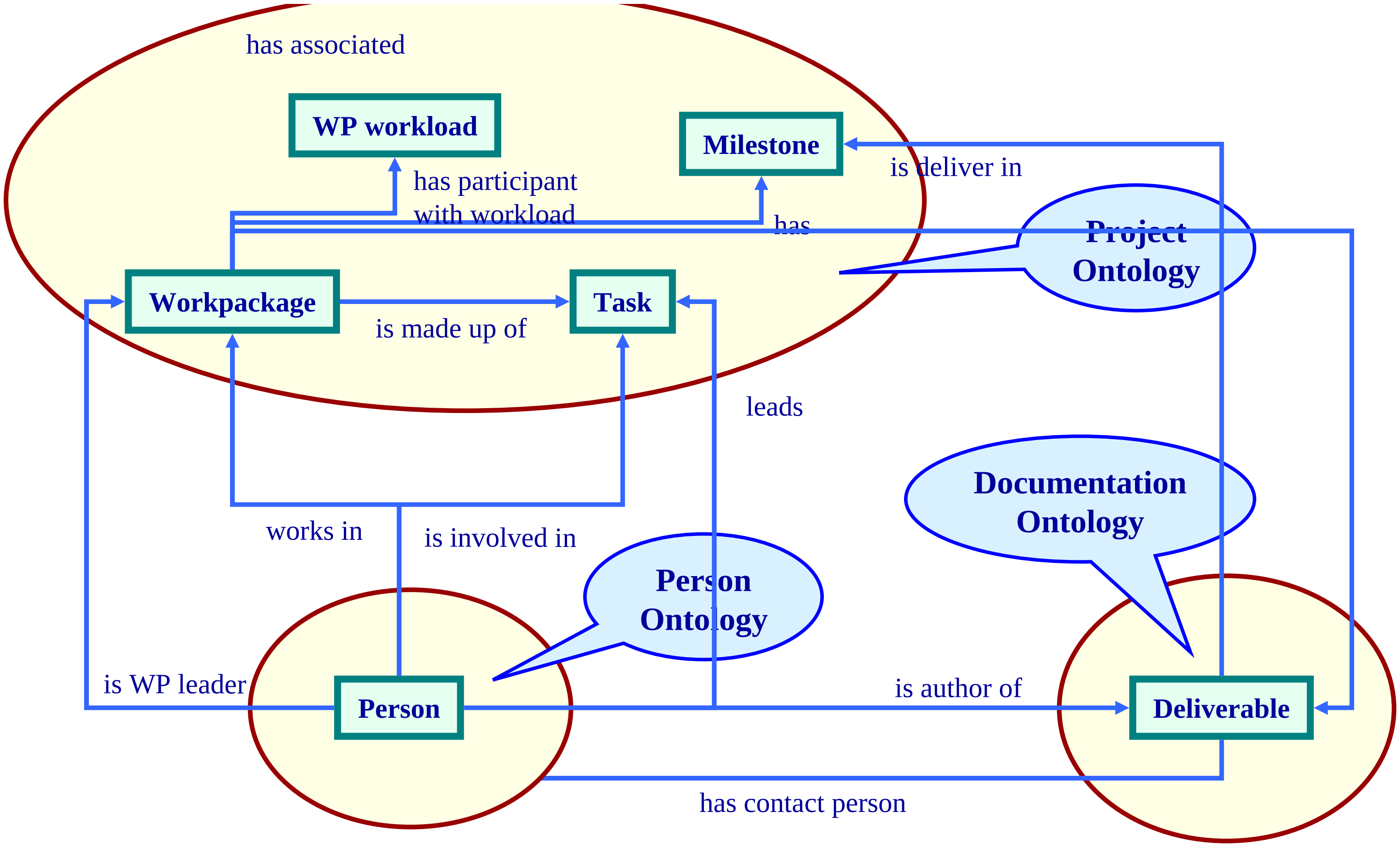
Event Ontology

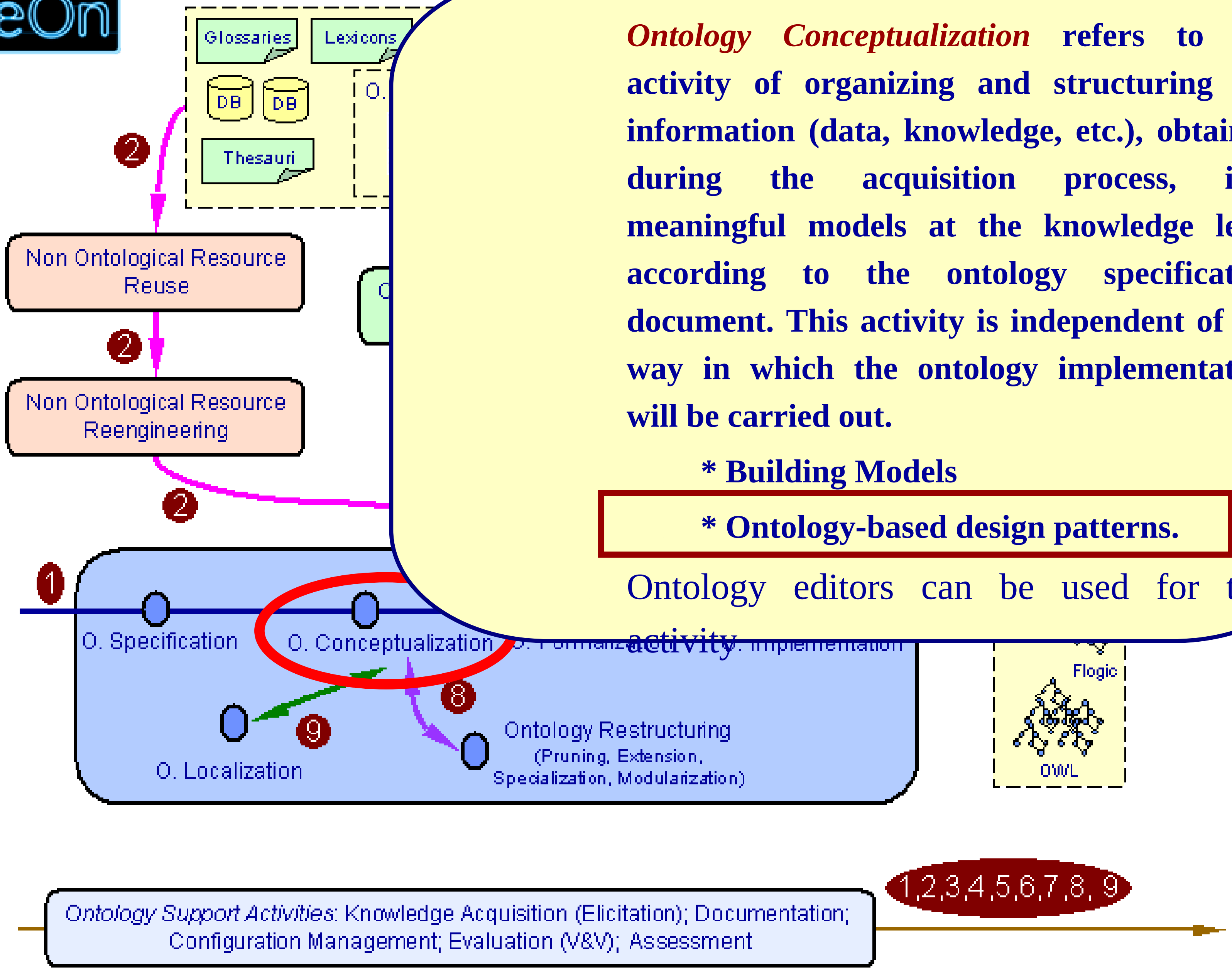


Documentation Ontology



Relationships between Person, Project and Documentation





Ontology Support Activities: Knowledge Acquisition (Elicitation); Documentation; Configuration Management; Evaluation (V&V); Assessment

***Ontology Conceptualization* refers to the activity of organizing and structuring the information (data, knowledge, etc.), obtained during the acquisition process, into meaningful models at the knowledge level according to the ontology specification document. This activity is independent of the way in which the ontology implementation will be carried out.**

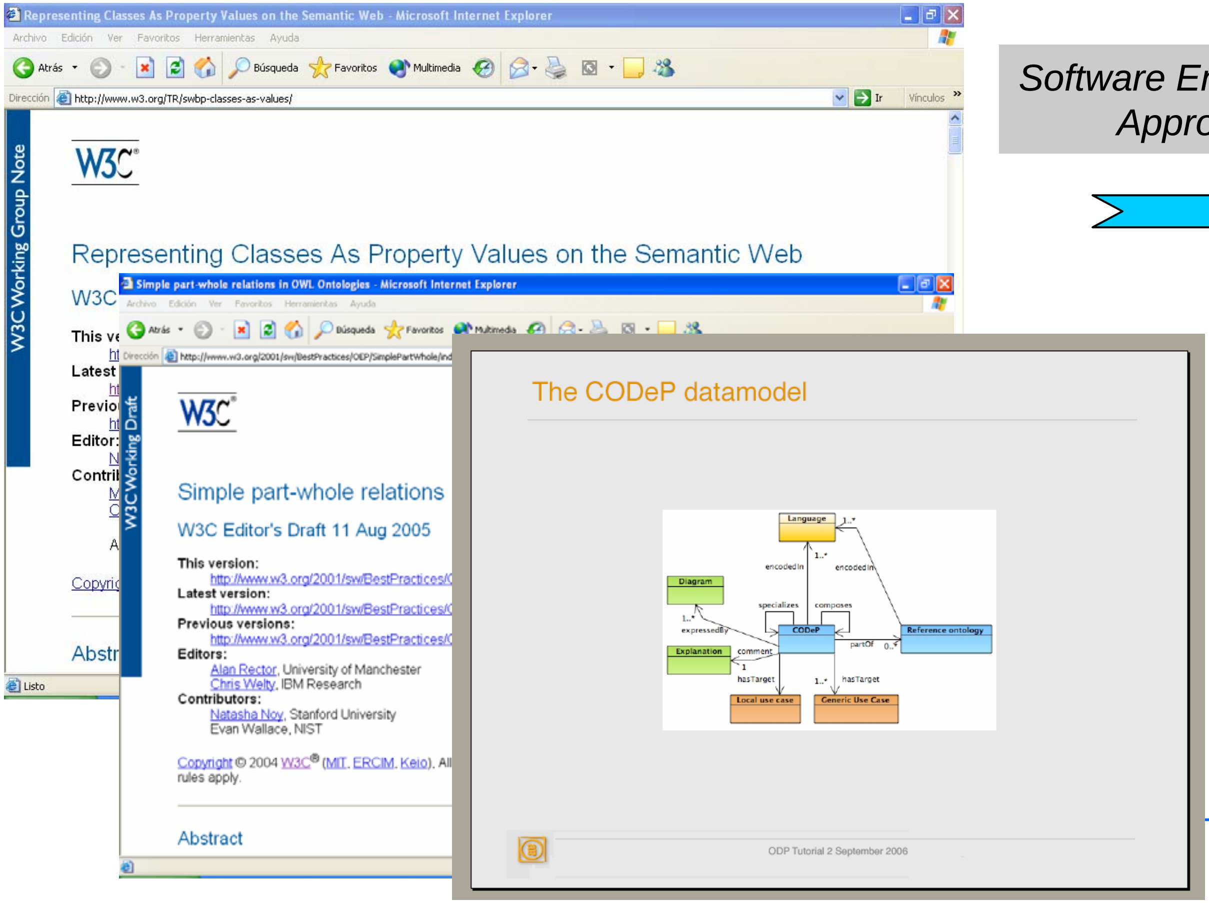
*** Building Models**

*** Ontology-based design patterns.**

Ontology editors can be used for this activity.

Ontology Design Patterns for Modelling

- **Pattern** is something proposed for imitation.
- **Design Pattern** refers to shared guidelines that help solve design problems.



Representing Classes As Property Values on the Semantic Web - Microsoft Internet Explorer

http://www.w3.org/TR/swbp-classes-as-values/

W3C Working Group Note

Representing Classes As Property Values on the Semantic Web

W3C

This ve

Latest

Previo

Editor:

Contri

Copyri

Abstr

Simple part-whole relations in OWL Ontologies - Microsoft Internet Explorer

http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/ind

W3C Working Draft

Simple part-whole relations

W3C Editor's Draft 11 Aug 2005

This version:
http://www.w3.org/2001/sw/BestPractices/O

Latest version:
http://www.w3.org/2001/sw/BestPractices/O

Previous versions:
http://www.w3.org/2001/sw/BestPractices/O

Editors:
Alan Rector, University of Manchester
Chris Welty, IBM Research

Contributors:
Natasha Noy, Stanford University
Evan Wallace, NIST

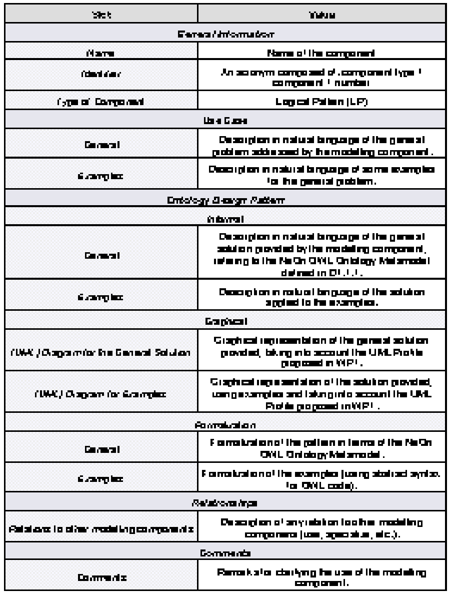
Copyright © 2004 W3C® (MIT, ERCIM, Keio), All rules apply.

Abstract

The CODeP datamodel

ODP Tutorial 2 September 2006

Software Engineering Approach



Slot	Value
General Information	
Name	Name of the component
Identifier	An acronym composed of component type + component + number
Type of Component	Logical Pattern (LP)
Use Case	
General	Description in natural language of the general problem addressed by the modeling component.
Examples	Description in natural language of some examples for the general problem.
Ontology Design Pattern	
Informal	
General	Description in natural language of the general solution provided by the modeling component, referring to the NeOn OWL Ontology Metamodel defined in D1.1.1.
Examples	Description in natural language of the solution applied to the examples.
Graphical	
(UML) Diagram for the General Solution	Graphical representation of the general solution provided, taking into account the UML Profile proposed in WP1.
(UML) Diagram for Examples	Graphical representation of the solution provided, using examples and taking into account the UML Profile proposed in WP1.
Formalization	
General	Formalization of the pattern in terms of the NeOn OWL Ontology Metamodel.
Examples	Formalization of the examples (using abstract syntax for OWL code).
Relationships	
Relations to other modeling components	Description of any relation to other modeling components (uses, specializes, etc.).
Comments	
Comments	Remarks for clarifying the use of the modeling component.

Inventory of Patterns. General Template

- *General Information*:
  - Name
  - Identifier
  - Ontology modelling component type (LP, AP and CP)
- *Use Case*, or problem to be addressed.
- *Ontology Design Pattern*, or proposed solution in different formats.
- *Relations to other ontology model components*. This slot is optional.
- *Comments*. This slot is also optional.

Logical Patterns. Definition

- **Elements of the OWL module from the NeOn metamodel or compositions of those elements**.
- Content-independent
  - Expressed with only a logical vocabulary (OWL namespace)
- Solve logical modelling problems
- Affect only a specific and delimited part of the ontology
- Examples:
  - Primitive and defined class, subClassOf relation, n-ary relation, etc.

18 Logical Patterns and Template

- LP for Modelling a Primitive Class
- LP for Modelling a Defined Class
- LP for Modelling a SubClassOf Relation
- LP for Modelling Multiple Inheritance between Class
- LP for Modelling an Equivalence Relation between C
- LP for Modelling an Object Property
- LP for Modelling a SubPropertyOf Relation
- LP for Modelling a Datatype Property
- LP for Modelling an Existential Restriction
- LP for Modelling a Universal Restriction
- LP for Modelling a UnionOf Relation
- LP for Modelling an Individual
- LP for Modelling Disjoint Classes
- LP for Modelling Exhaustive Classes
- **LP for Modelling N-ary Relation**
  - **Introducing a new class for the relation**
  - Using lists for arguments in the relation
- LP for Modelling Specified Values
  - Values as sets of individuals
  - Values as subclasses partitioning a 'feature'

Slot	Value
General Information	
Name	Name of the component
Identifier	An acronym composed of: component type + component + number
Type of Component	Logical Pattern (LP)
Use Case	
General	Description in natural language of the general problem addressed by the modelling component.
Examples	Description in natural language of some examples for the general problem.
Ontology Design Pattern	
Informal	
General	Description in natural language of the general solution provided by the modelling component, refering to the NeOn OWL Ontology Metamodel defined in D1.1.1.
Examples	Description in natural language of the solution applied to the examples.
Graphical	
(UML) Diagram for the General Solution	Graphical representation of the general solution provided, taking into account the UML Profile proposed in WP1.
(UML) Diagram for Examples	Graphical representation of the solution provided, using examples and taking into account the UML Profile proposed in WP1.
Formalization	
General	Formalization of the pattern in terms of the NeOn OWL Ontology Metamodel.
Examples	Formalization of the examples (using abstract syntax for OWL code).
Relationships	
Relations to other modelling components	Description of any relation to other modelling componens (use, specialize, etc.).
Comments	
Comments	Remarks for clarifying the use of the modelling component.

Example of a Logical Pattern: N-Ary Relations

- In Semantic Web languages such as RDF and OWL, a property is a binary relation. In some cases, the natural and convenient way to represent certain situations is to link an individual to more than just one individual or value by means of **n-ary relations**.
- One solution recommended by the W3C Semantic Web Best Practices and Deployment Working Group (SWBPD) is to introduce **a new class for the relation**.
- The solution resides in creating a new class and *n* new object properties to represent an n-ary relation. An instance of the relation linking the *n* individuals is then an instance of this class.

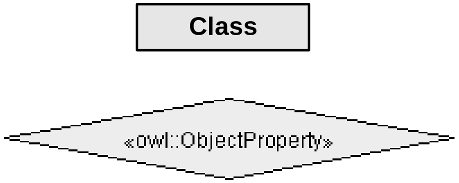
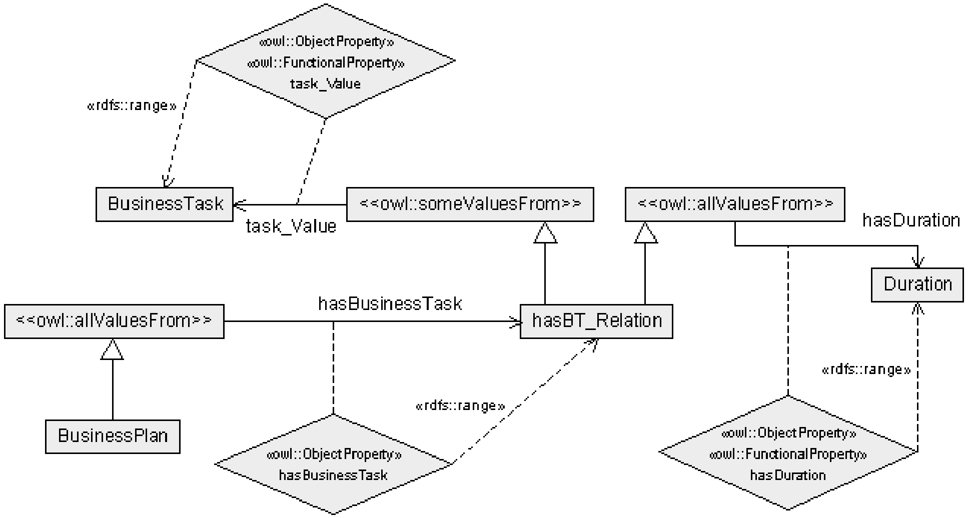
General Information	
Name	N-ary Relation: New Class
Identifier	LP-NR -01
Type of Component	Logical Pattern (LP)

Example of a Logical Pattern: N-Ary Relations

Use Case	
General	Express that: ▪ A binary relationship really needs a further argument. ▪ Two binary relationships always go together and should be represented as one n-ary relation. ▪ A relationship is really amongst several things.
Examples	Suppose that someone wants to express that ‘business plans’ have ‘business tasks’ with a concrete ‘duration’.

Ontology Design Pattern	
Informal	
General	Create a new class and *n* new object properties. Therefore, instantiate the classes Class and ObjectProperty.
Examples	Create the classes ‘BusinessPlan’, ‘BusinessTask’, ‘hasBT_Relation’, and ‘Duration’. In the definition of the class ‘BusinessPlan’, specify an object property ‘hasBusinessTask’ with the range restriction going to ‘hasBT_Relation’ class. Define ‘task_Value’ and ‘hasDuration’ as functional object properties. In the definition of the class ‘hasBT_Relation’, specify two object properties ‘task_Value’ and ‘hasDuration’ with the range restriction going to the classes ‘BusinessTask’ and ‘Duration’, respectively.

Example of a Logical Pattern: N-Ary Relations

Ontology Design Pattern	
Graphical	
(UML) Diagram for the General Solution	
(UML) Diagram for Examples	

Example of a Logical Pattern: N-Ary Relations

Ontology Design Pattern	
Formalization	

General	Class(Class partial OntologyElement)
	Class(Property partial OntologyElement)
	Class(ObjectProperty partial Property)

Examples	Class(BusinessPlan partial restriction(hasBusinessTask allValuesFrom(hBT_Relation)) owl:Thing)
	Class(hBT_Relation partial owl:Thing restriction(task_Value someValuesFrom(BusinessTask)) restriction(hasDuration allValuesFrom(Duration)))
	Class(BusinessTask partial owl:Thing)
	Class(Duration partial owl:Thing)
	ObjectProperty(task_Value Functional domain(owl:Thing))
	ObjectProperty(hasDuration Functional domain(owl:Thing))

Relationships	
Relations to other modelling components	Possible use of this LP in APs and CPs.

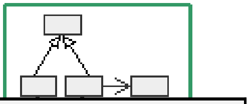
Architectural Patterns (APs). Definition

- Logical Patterns (LPs) or compositions of LPs
- Content-independent
- Solve architectural modeling problems
- Characterize the overall structure of the ontology
  - An AP expresses ‘how an ontology should look like’
- Examples:
  - Taxonomy, lightweight ontology, etc.

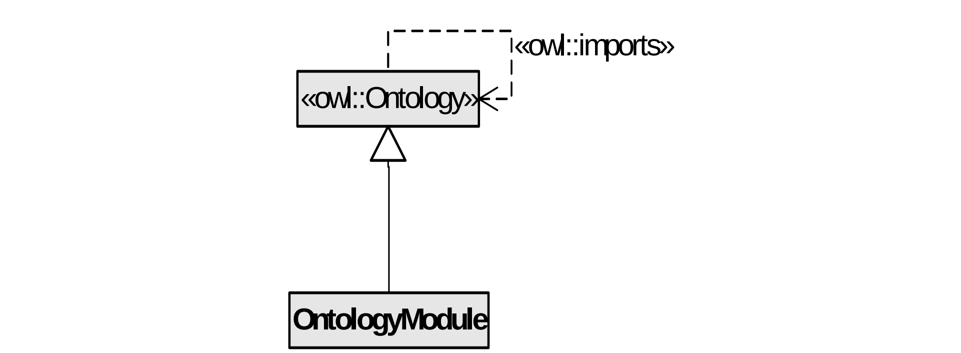
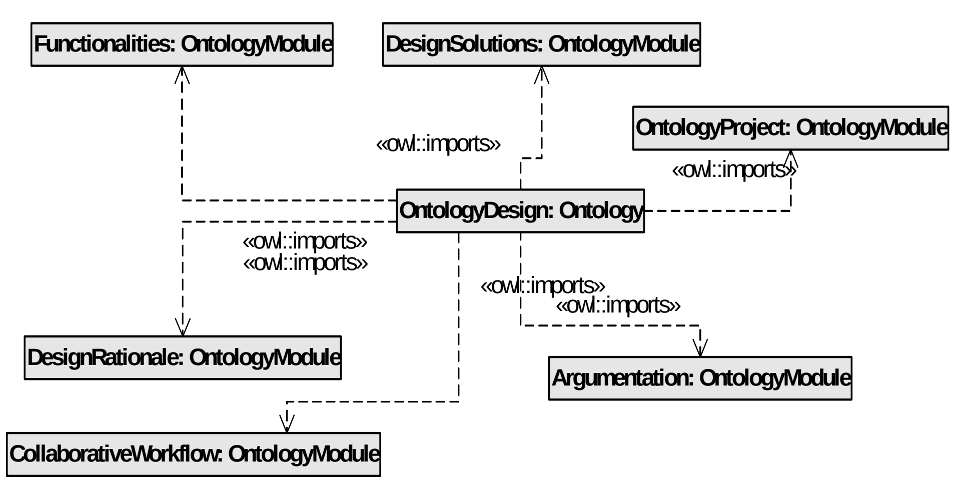
3 Architectural Patterns (APs)

- Taxonomy
  - A hierarchical structure of classes only related by subsumption relations.
- Lightweight ontology. Taxonomy + the following features:
  - A class can be related to other classes through the *disjointWith* relation.
  - Object and datatype properties can be defined and used to relate classes.
  - A specific domain and range can be associated with defined object and datatype properties.
- Modular architecture
  - Structuring an ontology as a configuration of components, each having its own identity based on some design criteria.
  - When an ontology is committed to a huge domain of knowledge, a good practice is to decompose the domain into smaller subdomains which address simpler tasks. Each subdomain can be then encoded in an ontology module, so as to provide the whole ontology with a modular architecture.

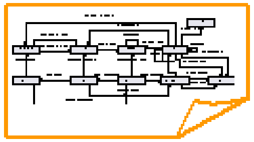
Architectural Patterns (APs) Template and Example



Slot	Value
Ge	
Name	
Identifier	
Type of Component	
General	
Examples	
Ontol	
General	
Examples	
(UML) Diagram for the General Solution	
(UML) Diagram for Examples	
Relations to other modelling components	componens (use, specialize, etc.).
Comments	
Comments	Remarks for clarifying the use of the modelling component.

Graphical	
(UML) Diagram for the General Solution	
(UML) Diagram for the Example Solution	 Functionalities: OntologyModule DesignSolutions: OntologyModule «owl::imports» OntologyProject: OntologyModule «owl::imports» OntologyDesign: Ontology «owl::imports» «owl::imports» «owl::imports» «owl::imports» DesignRationale: OntologyModule Argumentation: OntologyModule CollaborativeWorkflow: OntologyModule

Content Design Patterns (CPs). Definition



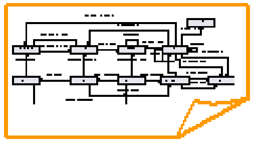
- **Instances of LPs or compositions of LPs.**
- Domain-dependent
  - – Expressed with a domain specific (non logical) vocabulary
- Solve domain modelling problems
- Affect the specific part of the ontology dealing with the domain modelling problem
- Examples:
  - – Plans, Medical Guidelines, Sales Order, etc.

6 Content Patterns (CPs) and Template

- **Participation Pattern**
- Description-Situation Pattern
- Role-Task Pattern
- Plan-Execution Pattern
- Simple Part-Whole Relation Pattern
  - Modelling a part-whole relation
  - Representing a part-whole class hierar

Slot	Value
General Information	
Name	Name of the component
Identifier	An acronym composed of: component type + component + number
Type of Component	Content Pattern (CP)
Use Case	
General	Description in natural language of the general problem addressed by the modelling component.
Examples	Description in natural language of some examples for the general problem.
Ontology Design Pattern	
Informal	
General	Description in natural language of the general solution provided by the modelling component, refering to the NeOn OWL Ontology Metamodel defined in D1.1.1. In this case we focus on a generic domain.
Examples	Description in natural language of the solution provided using examples. In this case we focus on a specific domain. This could be optional.
Graphical	
(UML) Diagram for the General Solution	Graphical representation of the general solution provided, taking into account the UML Profile proposed in WP1.
(UML) Diagram for Examples	Graphical representation of the solution provided, using examples and taking into account the UML Profile proposed in WP1. This could be optional.
Formalization	
General	Formalization of the pattern in terms of the most general classes and properties in OWL abstract syntax.
Examples	Formalization of specialized solution for the examples (using abstract syntax for OWL code). This could be optional.
Relationships	
Relations to other modelling components	Description of any relation to other modelling componens (use, specialize, etc.).
Comments	
Comments	Remarks for clarifying the use of the modelling component.

Example of Content Pattern: Participation

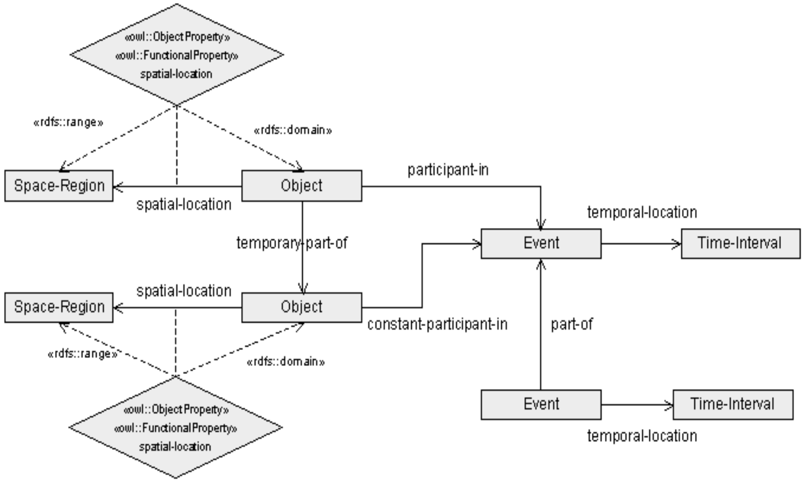


- This *NeOn Ontology Modelling Component* represents **participation** at spatio-temporal location .

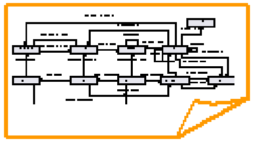
General Information	
Name	Participation
Identifier	CP-PA-01
Type of Component	Content Pattern (CP)

Use Case	
General	Express that objects take part in events.
Examples	Suppose that someones wants to represent people which take part in an international conference.

Example of Content Pattern: Participation

Ontology Design Pattern	
Informal	
General	The pattern consists in a 'participant-in' relation between *objects* and *events*, and assumes a time indexing for it. Time indexing is provided by the temporal location of the event at a *time interval*, while the respective spatial location at a *space region* is provided by the participating object. The pattern should instantiate the classes Class and ObjectProperty.
Graphical	
(UML) Diagram for the General Solution	 «owl::ObjectProperty» «owl::FunctionalProperty» spatial-location «rdfs::range» «rdfs::domain» Space-Region Object participant-in spatial-location temporal-location temporary-part-of Event Time-Interval spatial-location Space-Region Object constant-participant-in part-of «rdfs::range» «rdfs::domain» «owl::ObjectProperty» «owl::FunctionalProperty» spatial-location Event Time-Interval temporal-location *Note*: Diagram based on slides from the EKAW 2006 Tutorial about 'Ontology Design Patterns).

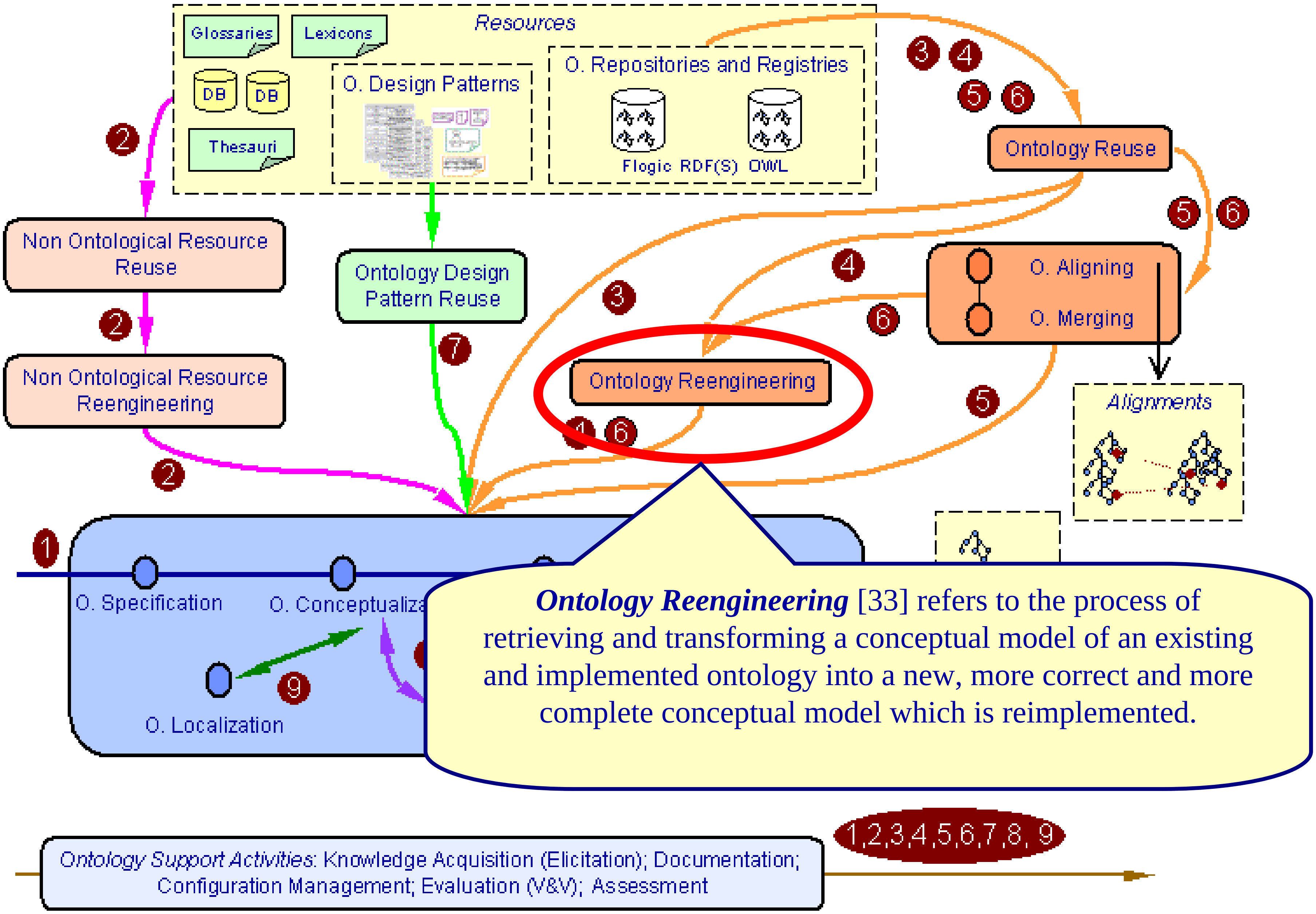
Example of Content Pattern: Participation



Ontology Design Pattern	
Formalization	

General	Class(Space-Region partial owl:Thing)
	ObjectProperty(spatial-location Functional domain(Object) range(Space-Region))
	Class(Object partial owl:Thing)
	ObjectProperty(temporary-part-of domain(Object) range(Object))
	ObjectProperty(participant-in domain(Object) range(Event))
	ObjectProperty(constant-participant-in domain(Object) range(Event))
	Class(Event partial owl:Thing)
	ObjectProperty(part-of domain(Event) range(Event))
	ObjectProperty(temporal-location domain(Event) range(Time-Interval))
	Class(Time-Interval partial owl:Thing)

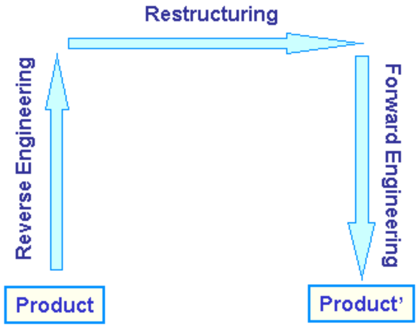
Relationships	
Relations to other modelling components	Relations to the following LPs: LP-DC / LP-PC and LP-OP.


Resources
Glossaries
Lexicons
DB
DB
Thesauri
O. Design Patterns
O. Repositories and Registries
Flogic RDF(S) OWL
Ontology Reuse
Non Ontological Resource Reuse
Ontology Design Pattern Reuse
Non Ontological Resource Reengineering
Ontology Reengineering
O. Aligning
O. Merging
Alignments
O. Specification
O. Conceptualiza
O. Localization
1
2
3
4
5
6
7
9
1,2,3,4,5,6,7,8, 9
Ontology Support Activities: Knowledge Acquisition (Elicitation); Documentation; Configuration Management; Evaluation (V&V); Assessment

Ontology Reengineering [33] refers to the process of retrieving and transforming a conceptual model of an existing and implemented ontology into a new, more correct and more complete conceptual model which is reimplemented.

Ontology Reengineering

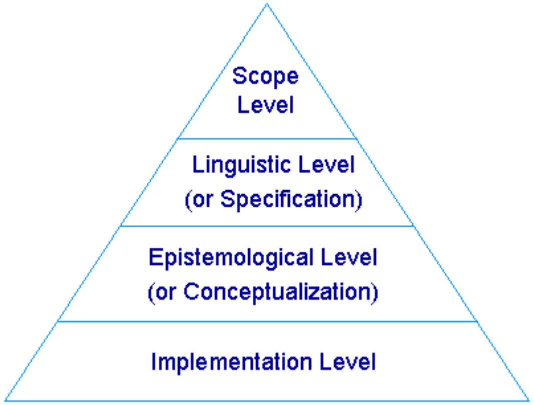
- **Ontology Reengineering** refers to the process of retrieving and transforming a conceptual model of an existing and implemented ontology into a new, more correct and more complete conceptual model which is reimplemented.
- **Ontology Forward Engineering** refers to the activity of outputting a new implementation of the ontology on the basis of the new conceptual model.
- **Ontology Restructuring** refers to the activity of correcting and reorganizing the knowledge contained in an initial conceptual model, and detecting missing knowledge.
- **Ontology Reverse Engineering** refers to the activity of outputting a possible conceptual model on the basis of the code in which the ontology is implemented.

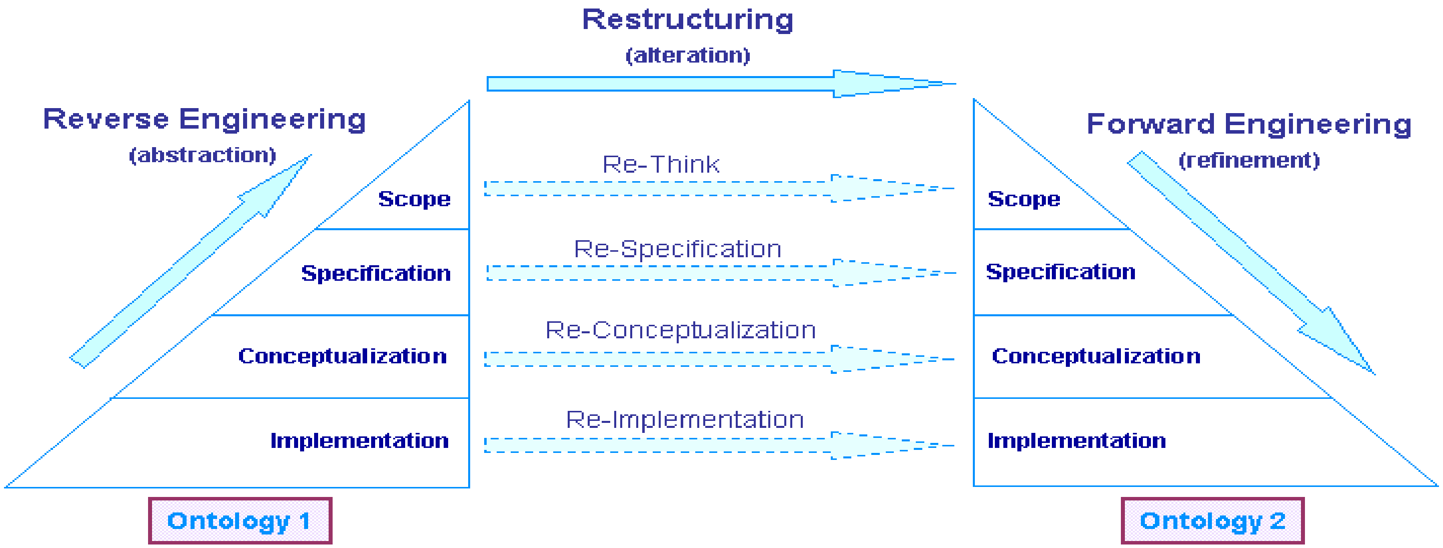


NeOn Glossary of Activities
(NeOn Deliverable D5.3.1)

Ontology Reengineering (II)

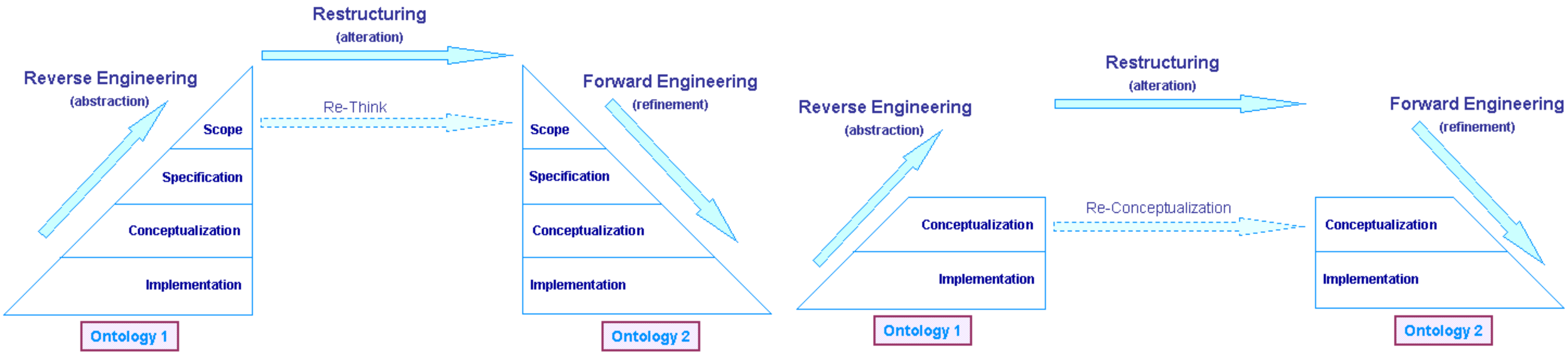
The **concept level of abstraction** that underlies the development process also underlies the reengineering process.


Scope Level
Linguistic Level (or Specification)
Epistemological Level (or Conceptualization)
Implementation Level



Ontology Reengineering (III)

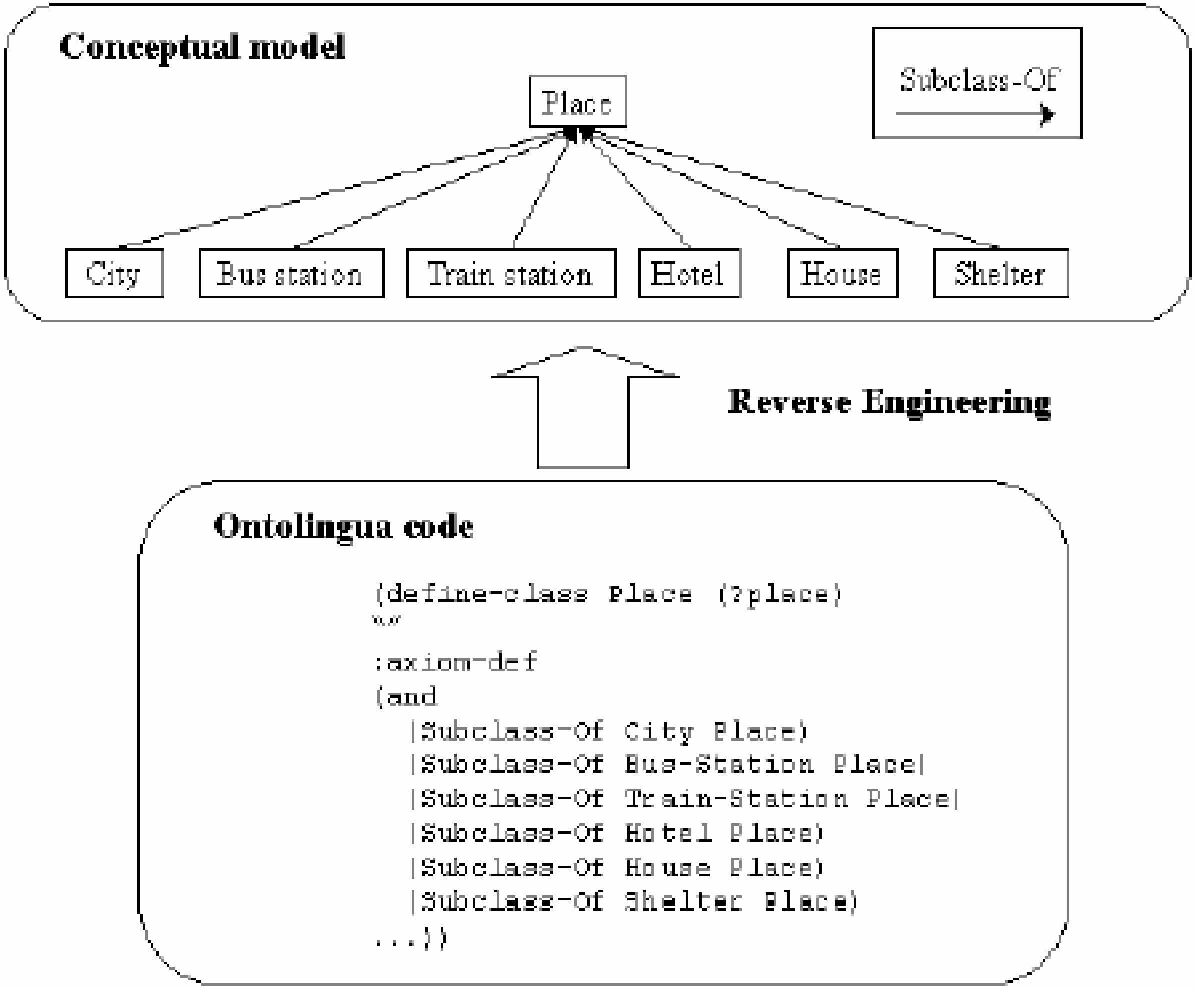
The **model** for ontology reengineering permit different ways, as for example:



Restructuring
(alteration)
Reverse Engineering
(abstraction)
Re-Think
Forward Engineering
(refinement)
Scope
Specification
Conceptualization
Implementation
Ontology 1
Scope
Specification
Conceptualization
Implementation
Ontology 2

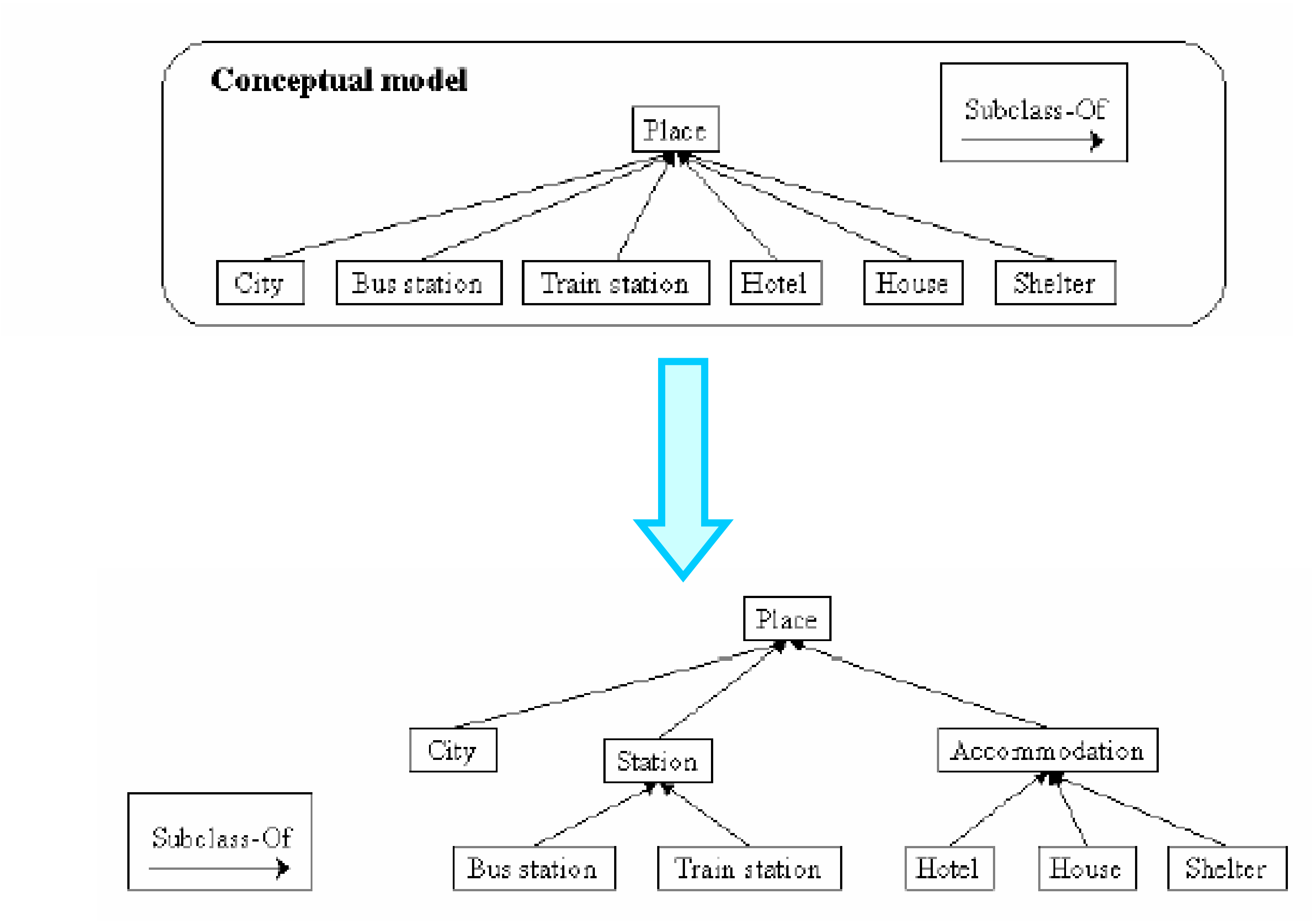
Restructuring
(alteration)
Reverse Engineering
(abstraction)
Re-Conceptualization
Forward Engineering
(refinement)
Conceptualization
Implementation
Ontology 1
Conceptualization
Implementation
Ontology 2

Reverse Engineering. Example



```
(define-class Place (?place)
""
:axiom-def
(and
   (Subclass-Of City Place)
   (Subclass-Of Bus-Station Place)
   (Subclass-Of Train-Station Place)
   (Subclass-Of Hotel Place)
   (Subclass-Of House Place)
   (Subclass-Of Shelter Place)
...))
```

Restructuring. Example

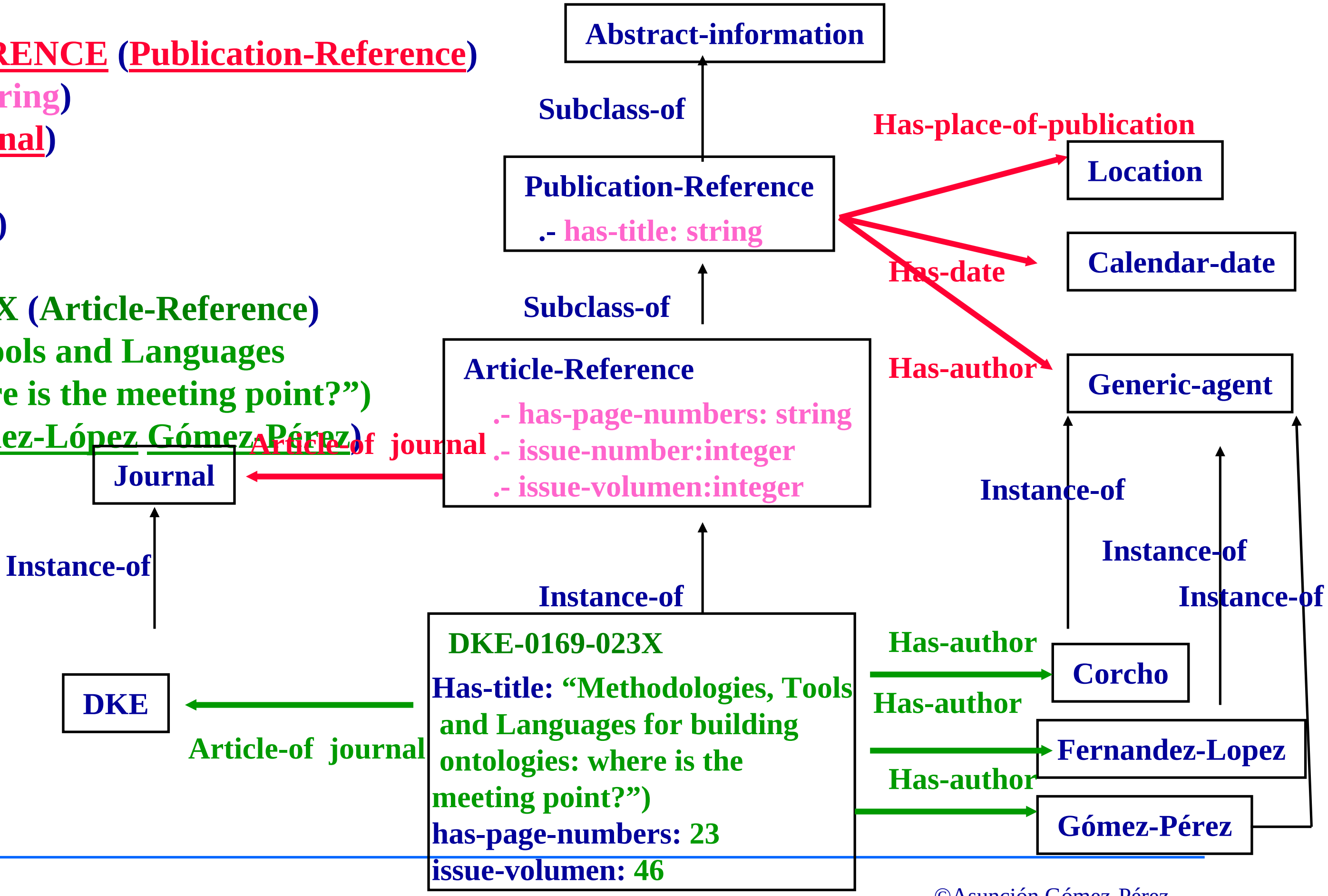


Reverse Engineering. Example 2

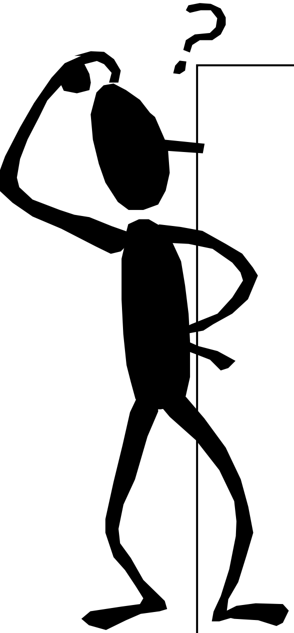
```
(def-class PUBLICATION-REFERENCE (abstract-information)
"we have decided that a publication reference is an intangible, abstract information"
((has-title :type string)
(has-author :type generic-agent)
(has-date :type calendar-date)
(has-place-of-publication :type location)))

(def-class ARTICLE-REFERENCE (Publication-Reference)
 ((has-page-numbers :type string)
(article-of-journal :type journal)
(issue-number :type integer)
(issue-volume :type integer)))

(def-instance DKE-0169-023X (Article-Reference)
(has-title "Methodologies, Tools and Languages
for building ontologies: where is the meeting point?")
(has-author Corcho Fernández-López Gómez-Pérez)
(has-date July-2003)
(has-page-numbers 23)
(article-of-journal DKE)
(issue-volume 46))
```

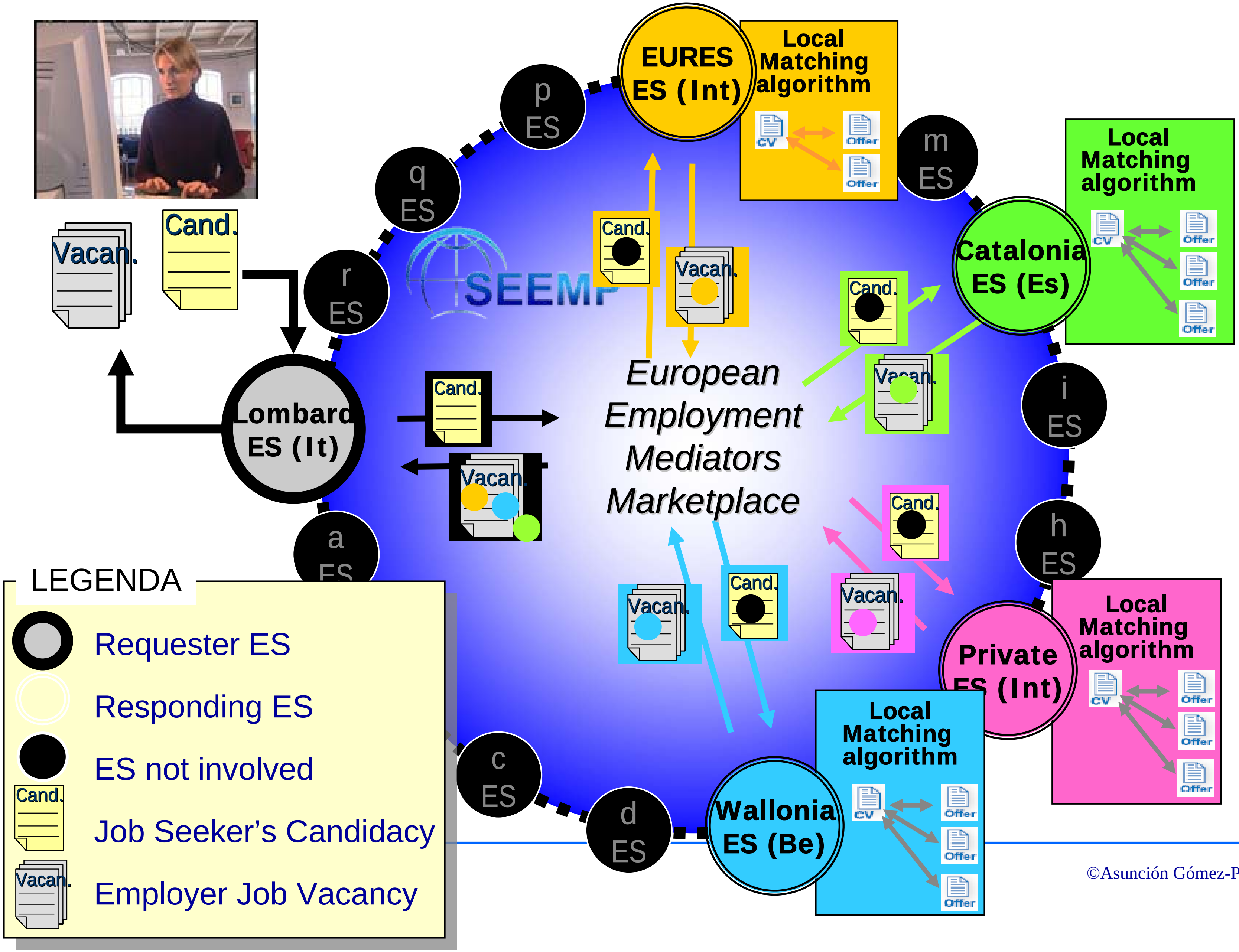


I want to build my ontology



- Which one are the activities involved in the ontology development process?
- Which one is the goal of each activity?
- When should I carry out each activity?
- Where is the relationship of one activity with the others?
- Where can I find ontologies with the goal of reusing them?
- How can I build the ontology for my application?
- Do I need a single ontology or an ontology network?
  - Example of building an ontology in the *Employment Mediators Marketplace*

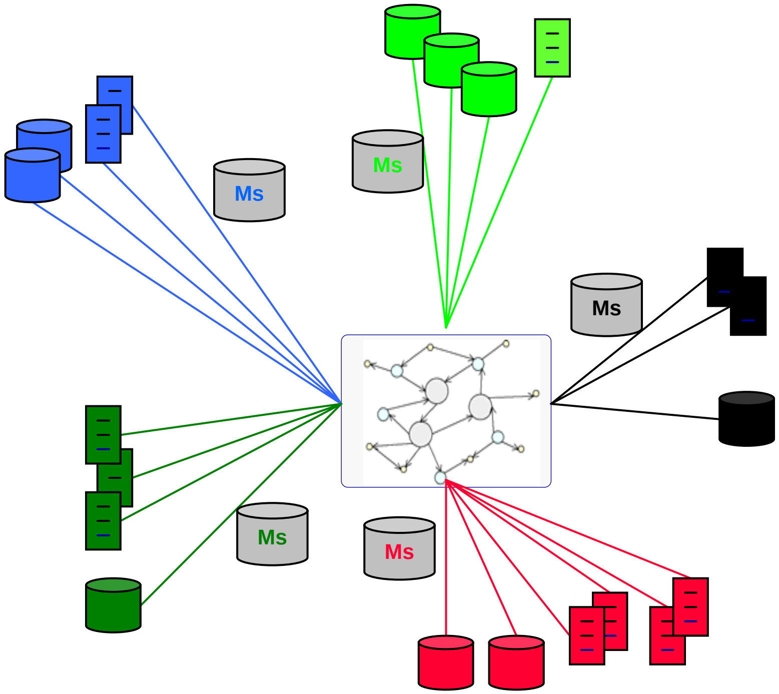
Helping Job Seekers on their way



LEGENDA

- ● Requester ES
- ○ Responding ES
- ● ES not involved
- Cand. Job Seeker's Candidacy
- Vacan. Employer Job Vacancy

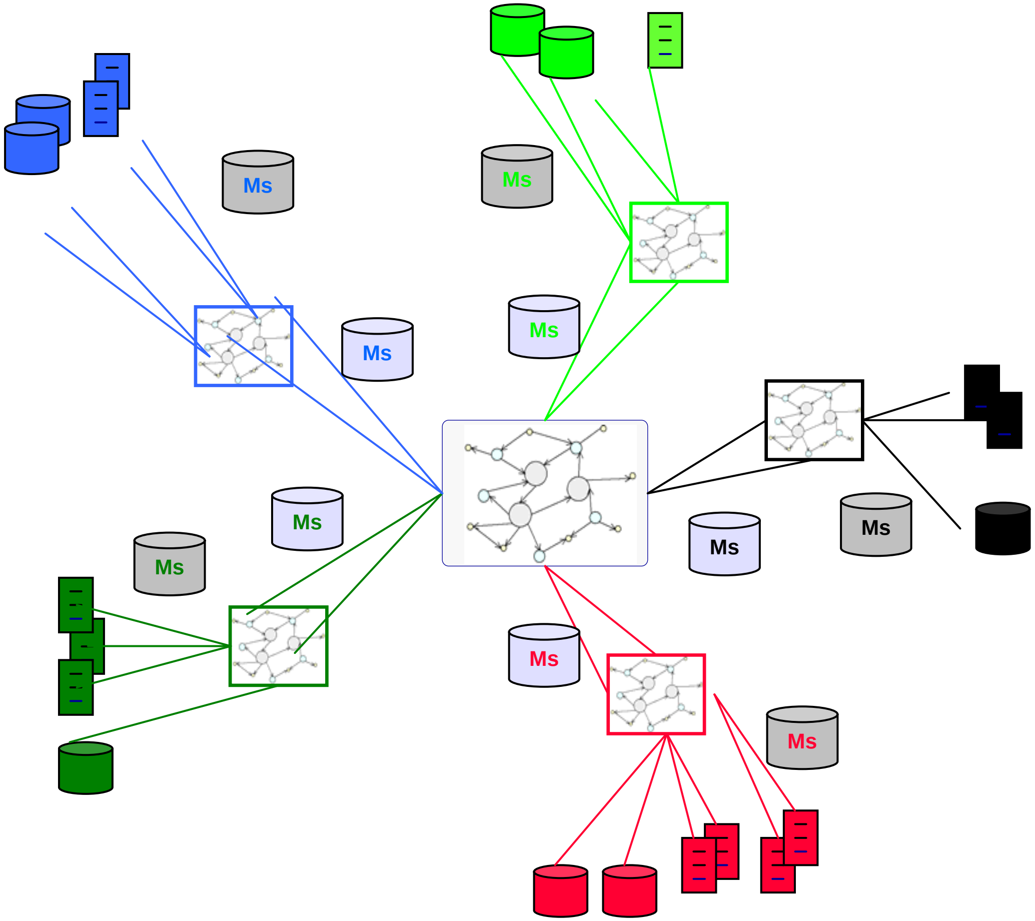
Centralized network of ontologies



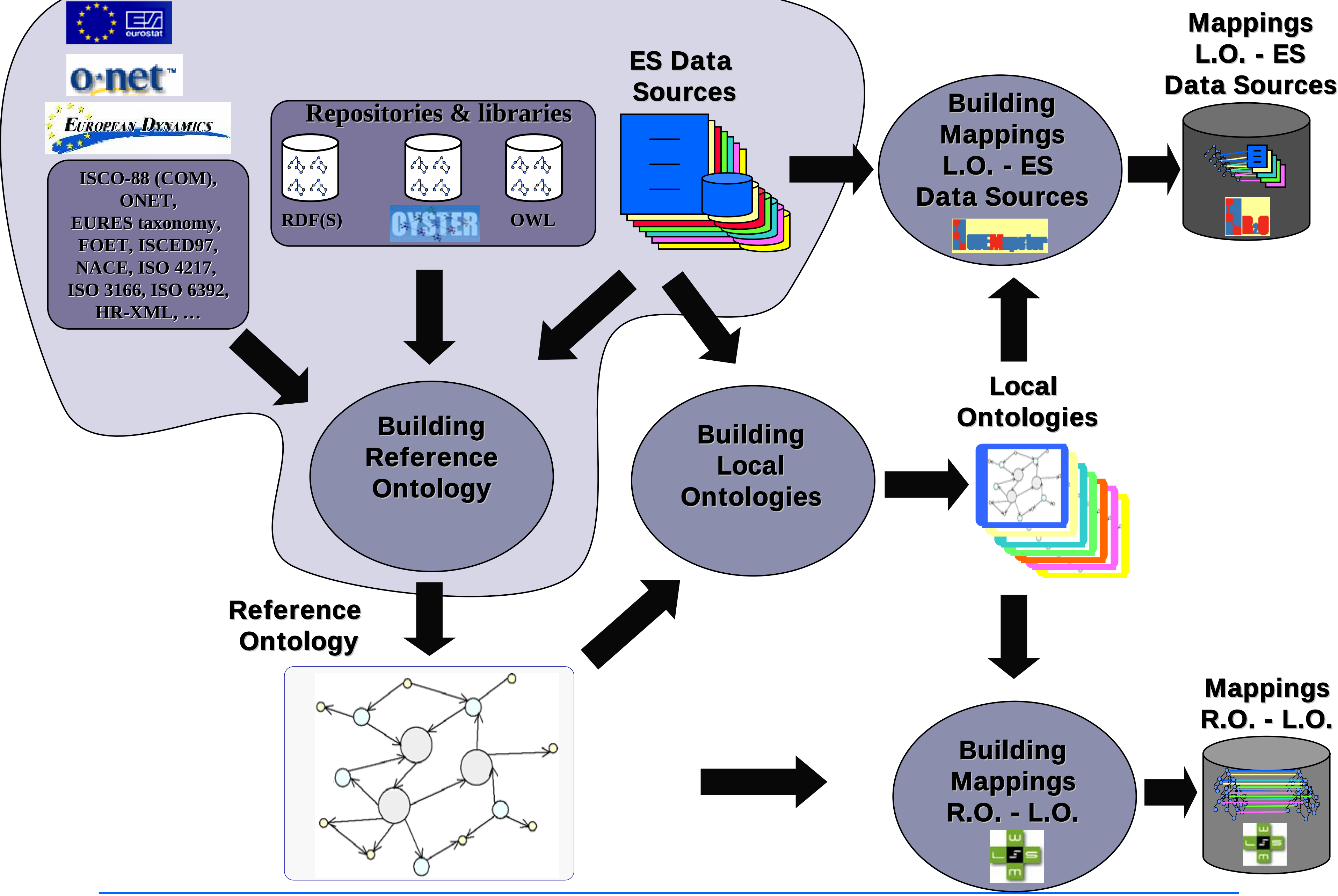
1. Build a reference ontology

2. Build mappings between the reference ontology and the data sources

Federated network of ontologies



1. Build a reference ontology for the domain
2. Build local ontologies
3. Build mappings between the core and local ontologies
4. Build mappings between the local ontologies and the data sources



ISCO-88 (COM), ONET, EURES taxonomy, FOET, ISCED97, NACE, ISO 4217, ISO 3166, ISO 6392, HR-XML, …

Repositories & libraries

RDF(S)

OWL

ES Data Sources

Building Mappings L.O. - ES Data Sources

Mappings L.O. - ES Data Sources

Building Reference Ontology

Building Local Ontologies

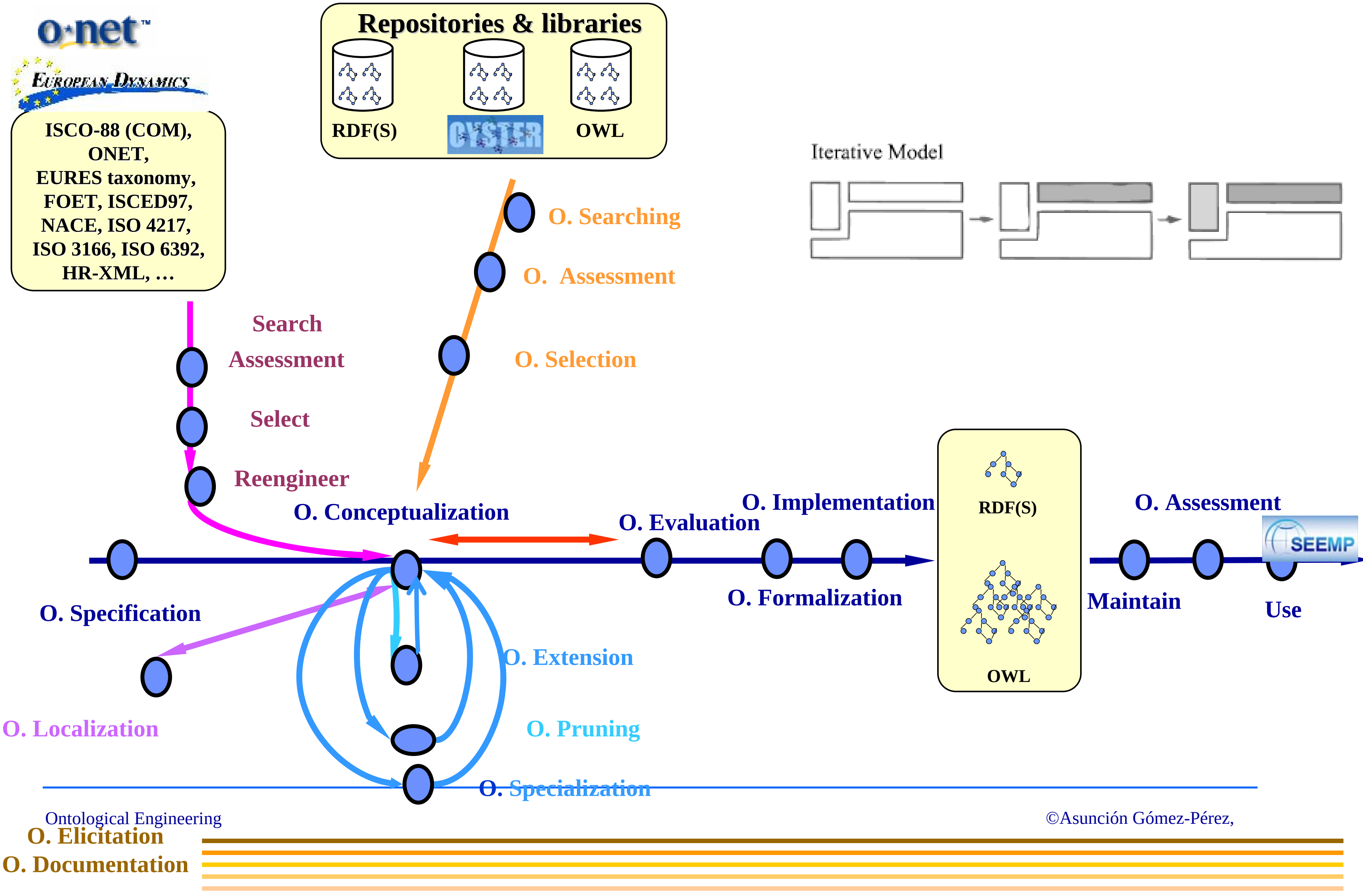
Local Ontologies

Reference Ontology

Building Mappings R.O. - L.O.

Mappings R.O. - L.O.

SEEMP Ontology
Network Life Cycle: Iterative model life cycle



ISCO-88 (COM), ONET, EURES taxonomy, FOET, ISCED97, NACE, ISO 4217, ISO 3166, ISO 6392, HR-XML, …

Repositories & libraries

RDF(S)

OWL

Iterative Model

Search

Assessment

Select

Reengineer

O. Searching

O. Assessment

O. Selection

O. Conceptualization

O. Evaluation

O. Implementation

O. Formalization

RDF(S)

OWL

O. Assessment

Maintain

Use

O. Specification

O. Localization

O. Extension

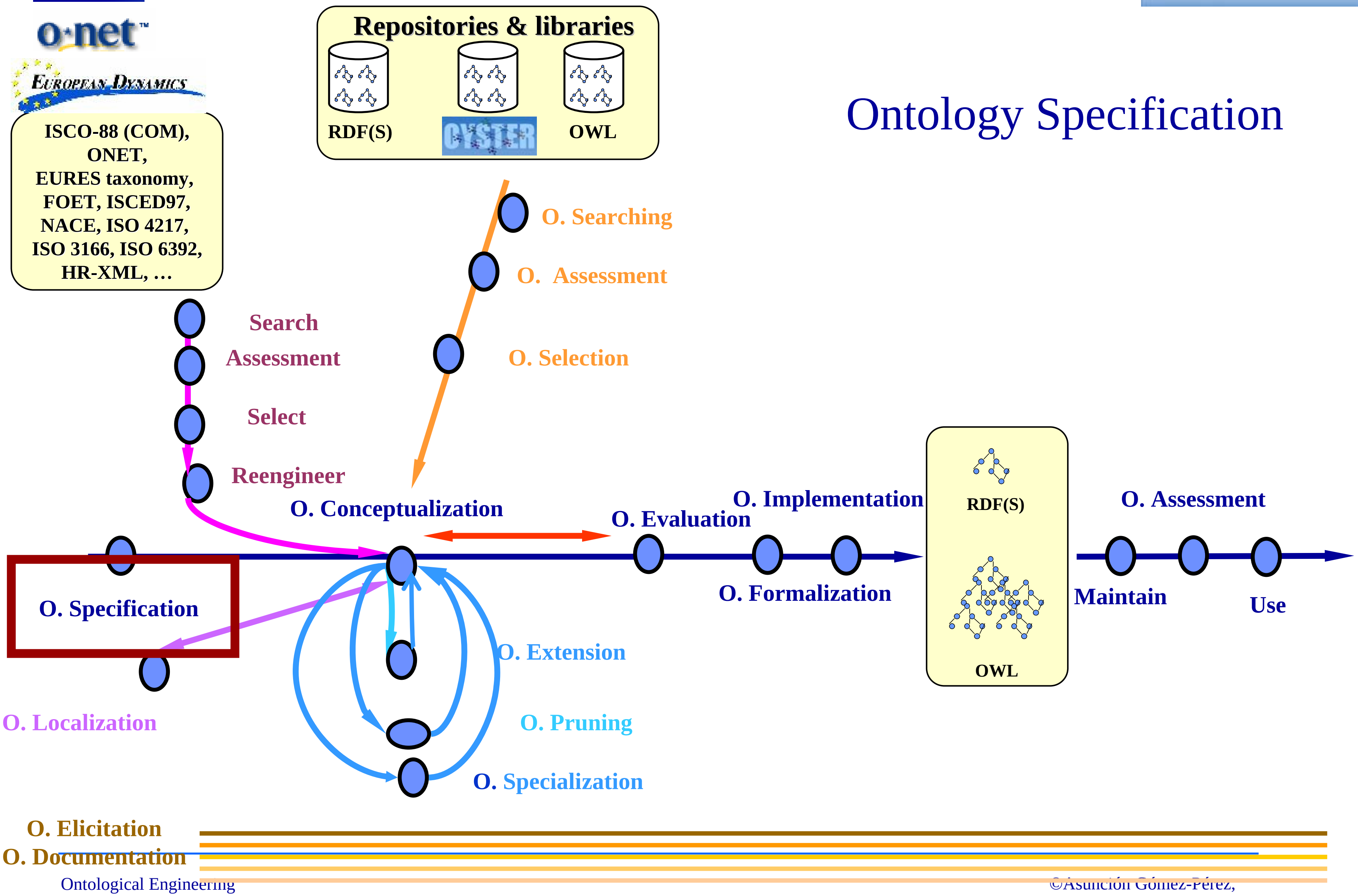
O. Pruning

O. Specialization

O. Elicitation

O. Documentation

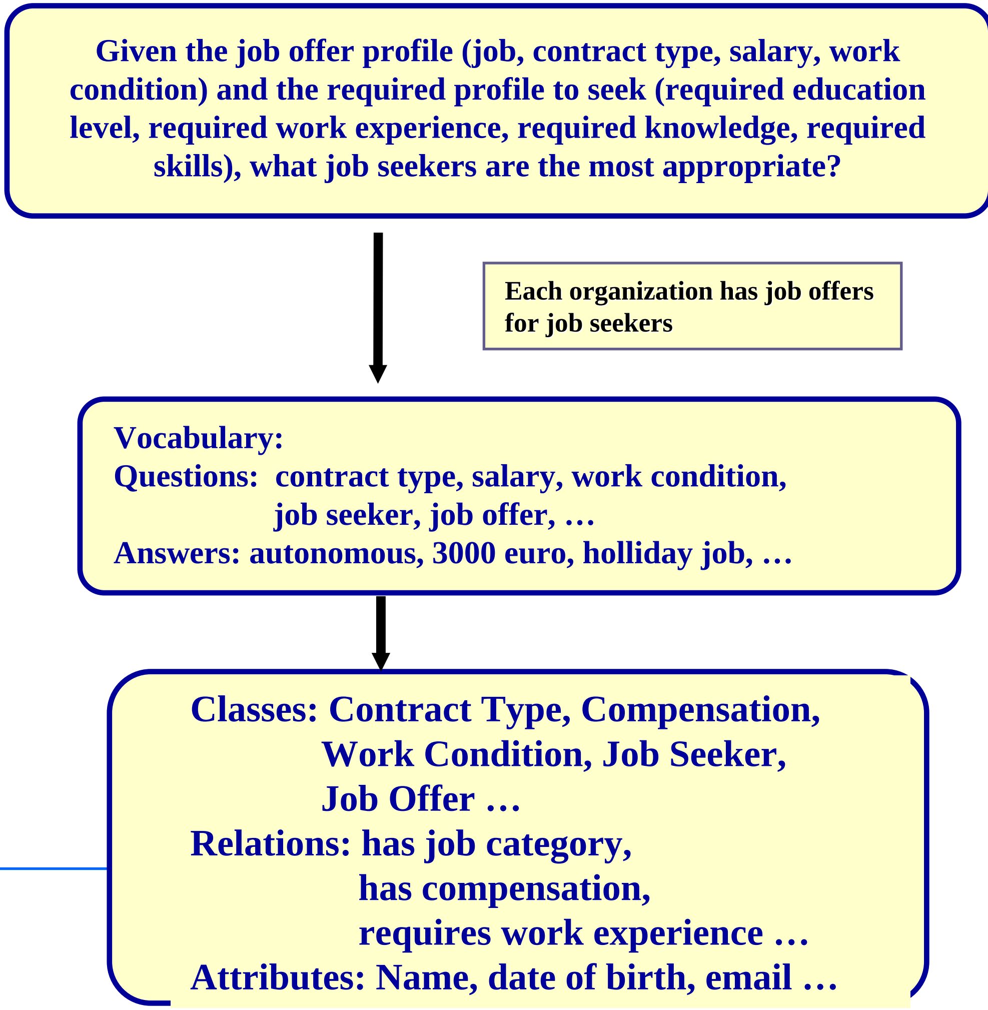
Ontology Specification



Ontology Specification

60 Competency questions grouped into 5 categories (modular approach)

- Job Seeker (12)
  - What is his/her education level?
- Job Offer (12)
  - What are the required skills for the job offer?
- Time and date management (7)
  - When the job seeker completed his/her first degree?
- Currencies (4)
  - The offered salary is given in US dollars?
- General (25)
  - Given the employer information, economic activity of the employer and the job offer profile (job, contract type, salary, work condition, contract duration), what job seekers are the most appropriate?



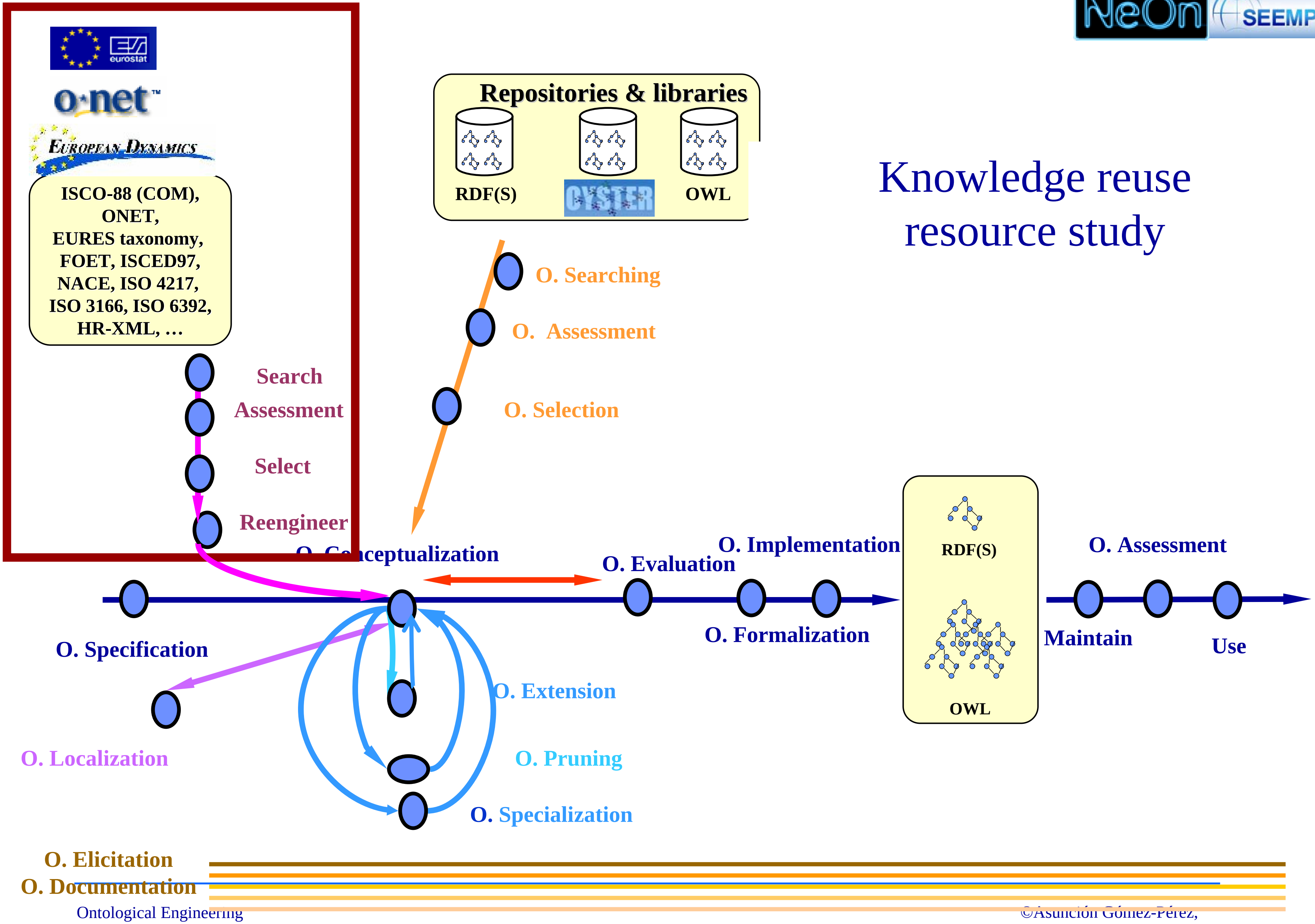
Given the job offer profile (job, contract type, salary, work condition) and the required profile to seek (required education level, required work experience, required knowledge, required skills), what job seekers are the most appropriate?

Each organization has job offers for job seekers

Vocabulary:
Questions: contract type, salary, work condition, job seeker, job offer, …
Answers: autonomous, 3000 euro, holliday job, …

Classes: Contract Type, Compensation, Work Condition, Job Seeker, Job Offer …
Relations: has job category, has compensation, requires work experience …
Attributes: Name, date of birth, email …

Knowledge reuse resource study



ISCO-88 (COM),
ONET,
EURES taxonomy,
FOET, ISCED97,
NACE, ISO 4217,
ISO 3166, ISO 6392,
HR-XML, …

Search

Assessment

Select

Reengineer

Repositories & libraries

RDF(S)

OYSTER

OWL

O. Searching

O. Assessment

O. Selection

O. Conceptualization

O. Evaluation

O. Implementation

O. Formalization

RDF(S)

OWL

O. Assessment

Maintain

Use

O. Specification

O. Localization

O. Extension

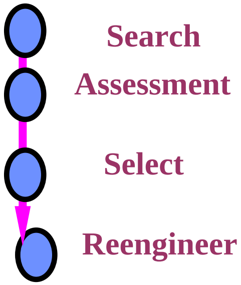
O. Pruning

O. Specialization

O. Elicitation

O. Documentation

Search and Assess Standards and Taxonomies



- *We select the most appropriate standards and taxonomies for:*
  - Occupation Classification
    ISCO-88 (COM), SOC, ISCO-88, ONET, Eures Taxonomy.
  - Classification of Economic Activities
    ISIC Rev. 3.1, NACE Rev. 1.1, NAICS
  - Apprenticeship classifications
    ISCED 97, FOET
  - Currency Classification
    ISO 4217
  - Geography Classification
    ISO 3166, Eures Taxonomy

Language Classification
ISO 6392, CEF

Driving License Classification
European Legislation

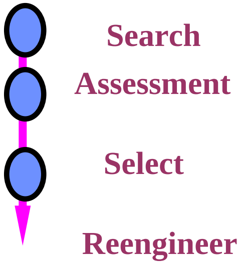
Skill Classification
Eures Taxonomy

Contract Types Classification
LE FOREM, Eures and BLL Classification

Work Condition Classification
LE FOREM, Eures and BLL Classification

Assessment activity: Matching terminology from Competency Questions against the Standards

Selection of Standards



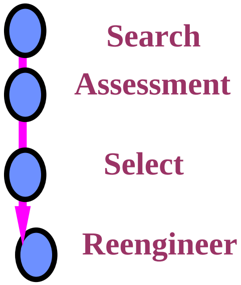
Reference Ontology shall be based on the international, European or de-facto industrial standards

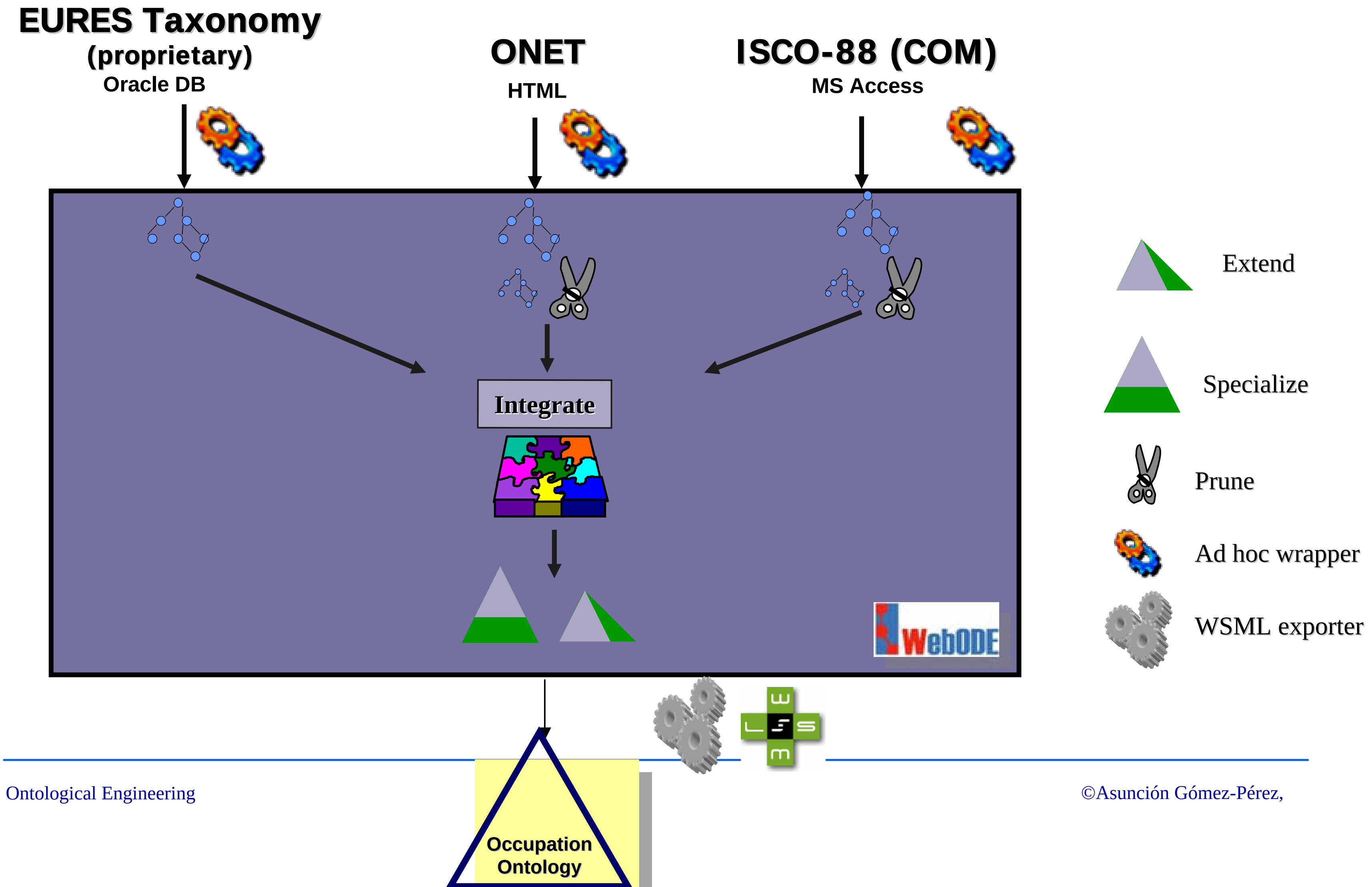
	Occupation Classification				Classification of Economic Activities			Apprenticeship Classification	
	SOC	ONET	ISCO-88	ISCO-88 (COM)	ISIC Rev. 3.1	NACE Rev. 1.1	NAICS	ISCED 97	FOET
The degree of coverage		☑	☑	☑	☑	☑			☑
The current European needs				☑		☑		☑	☑

Currency Classification	Geography Classification	Language Classification	Driving License
ISO 4217	ISO 3166	ISO 6392	Community Driving License

But, we need also proprietary taxonomies ...

Reengineering resources


Search
Assessment
Select
Reengineer


EURES Taxonomy
(proprietary)
Oracle DB

ONET
HTML

ISCO-88 (COM)
MS Access

Integrate

Extend

Specialize

Prune

Ad hoc wrapper

WSML exporter

Knowledge Resource Reengineering

○ Search Assessment

○ Select

○ Reengineer

ISO 3166-1 (XML)

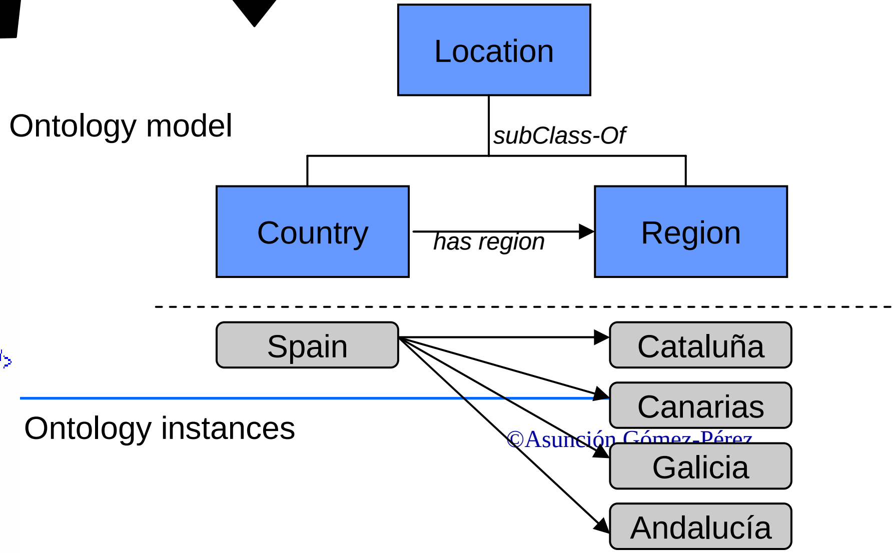
```
....

 <ISO_3166-1_Entry>
     <ISO_3166-1_Country_name>SPAIN</ISO_3166-1_Country_name>
     <ISO_3166-1_Alpha-2_Code_element>ES</ISO_3166-1_Alpha-
2_Code_element>
   </ISO_3166-1_Entry>
...
```

Regions Table (Eures Oracle DB)

N	ISO31661 Code	Region
100	ES	Cataluña
101	ES	Canarias
102	ES	Galicia
103	ES	Andalucia
104	ES	Navarra
105	ES	Asturias
106	ES	Baleares
107	ES	Murcia
108	ES	Aragon

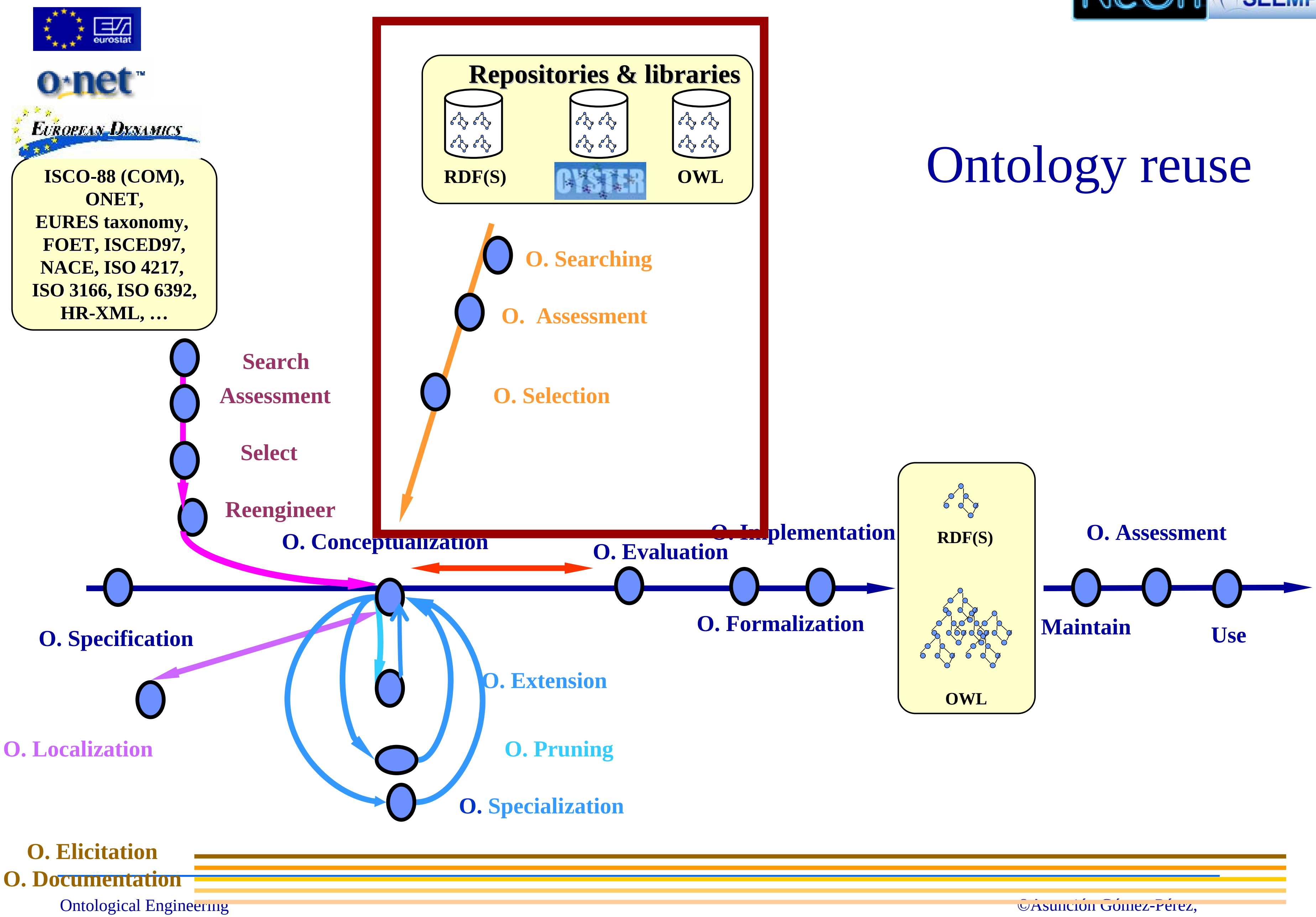
Excerpt of the Geography Ontology



©Asunción Gómez-Pérez

```
<rdf:Description rdf:about="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Country_SPAIN">
  <rdf:type rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Country"/>
  <GeoOnt:Code rdf:datatype="http://www.w3.org/2001/XMLSchema#string">ES</GeoOnt:Code>
  <GeoOnt:Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">SPAIN</GeoOnt:Name>
  <GeoOnt:is_located_in_Continent rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#EU_Europe"/>
  <GeoOnt:has_region_Region rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Catalunya"/>
  <GeoOnt:has_region_Region rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Canarias"/>
  <GeoOnt:has_region_Region rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Galicia"/>
  <GeoOnt:has_region_Region rdf:resource="webode://mccarthy.dia.fi.upm.es/Geography_Ontology#Andalucia"/>
</rdf:Description>
```

Ontology reuse



ISCO-88 (COM),
ONET,
EURES taxonomy,
FOET, ISCED97,
NACE, ISO 4217,
ISO 3166, ISO 6392,
HR-XML, …

Search
Assessment
Select
Reengineer

Repositories & libraries
RDF(S)
OYSTER
OWL

O. Searching
O. Assessment
O. Selection

O. Conceptualization
O. Evaluation
O. Implementation
O. Formalization
O. Specification
O. Localization
O. Extension
O. Pruning
O. Specialization

RDF(S)
OWL

O. Assessment
Maintain
Use

O. Elicitation
O. Documentation

Assessing Time Ontologies

O. Searching

O. Assessment

O. Selection

1. Identification of criteria for comparing the candidate set of temporal ontologies

- Time Points
- Time Interval
- Absolute and Relative Time
- Relations between time intervals
- Convex and non convex intervals

- Distinction between open and closed intervals
- Explicit modeling of proper intervals
- Concatenation of intervals
- Different temporal granularities

2. Assess all existing temporal ontologies against the criteria

	Cyc’s Upper Ontology ☑	Unrestricted Time Ontology ☑	Simple Time Ontology ☑	Reusable Time Ontology ☑	Kestrel Time Ontology ☑	SRI’s Time Ontology ☑	SUMO Time Ontology ☑	DAML Time Ontology ☑	AKT Time Ontology ☑
Time Points	☑				☑	☑	☑	☑	☑
Time Interval			☑	☑				☑	☑
Absolute and Relative Time					☑		☑	☑	
Relations between time intervals				☑				☑	
Convex and non convex intervals				☑			☑	☑	
Distinction between open and closed intervals								☑	
Explicit modeling of proper intervals								☑	
Concatenation of intervals	☑					☑	☑	☑	☑
Different temporal granularities		☑	☑	☑			☑	☑	
Provides axioms									

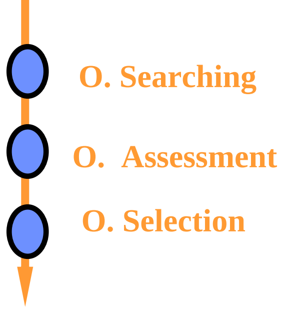
Process for assessing Time Ontologies (II)

3. Checking which temporal properties are needed for answering the Competency questions
   a. When the job seeker completed his/her first degree?
   b. Is the job seeker older than 30 years?
   c. How much time did the job seeker spend completing his/her first degree?
   d. How long is the duration of the contract?
   e. Which job offers were posted in last 24 hours?
   f. Which job offers were posted in last 7 days?
   g. Which job offers were posted in last month?
   h. Was the job seeker unemployed?
   i. Was the job seeker a student between 1995 and 2000?

Time Points	a
Time Interval	b, c
Absolute and Relative Time	a,d,f,g
Relations between time intervals	
Convex and non convex intervals	h
Distinction between open and closed intervals	a,d,f,g
Explicit modeling of proper intervals	i
Concatenation of intervals	
Different temporal granularities	a,d,f,g

4. Checking which temporal properties are needed for answering the Competency questions

The Time Ontology Selection



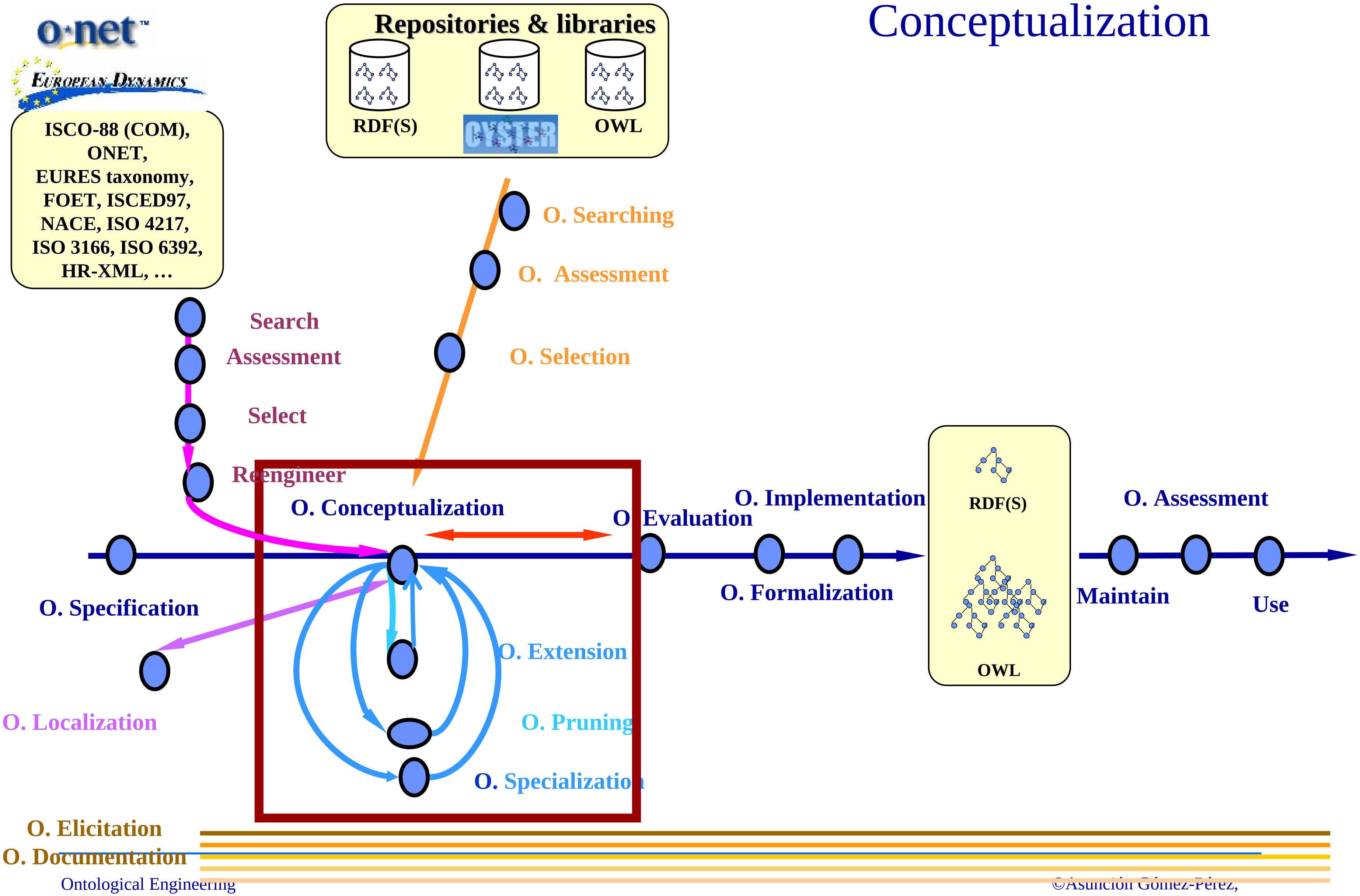
O. Searching

O. Assessment

O. Selection

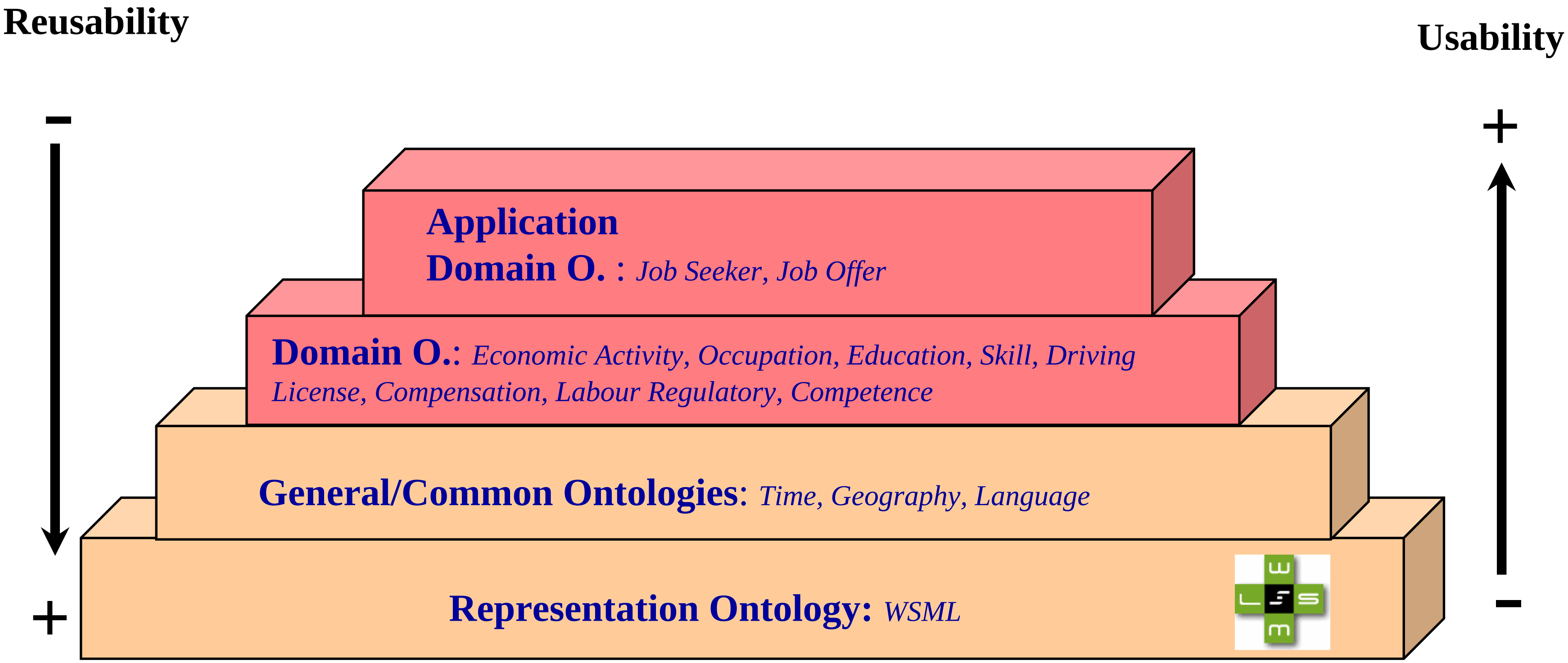
	Cyc's Upper Ontology	Unrestricted Time Ontology	Simple Time Ontology	Reusable Time Ontology	Kestrel Time Ontology	SRI's Time Ontology	SUMO Time Ontology	DAML Time Ontology	AKT Time Ontology
Time Points ●	☑	☑	☑	☑	☑	☑	☑	☑	☑
Time Interval ●	☑				☑	☑	☑	☑	☑
Absolute and Relative Time ●			☑	☑				☑	☑
Relations between time intervals					☑		☑	☑	
Convex and non convex intervals ●				☑				☑	
Distinction between open and closed intervals ●				☑			☑	☑	
Explicit modeling of proper intervals ●								☑	
Concatenation of intervals								☑	
Different temporal granularities	☑					☑	☑	☑	☑
Provides axioms ●		☑	☑	☑			☑	☑	

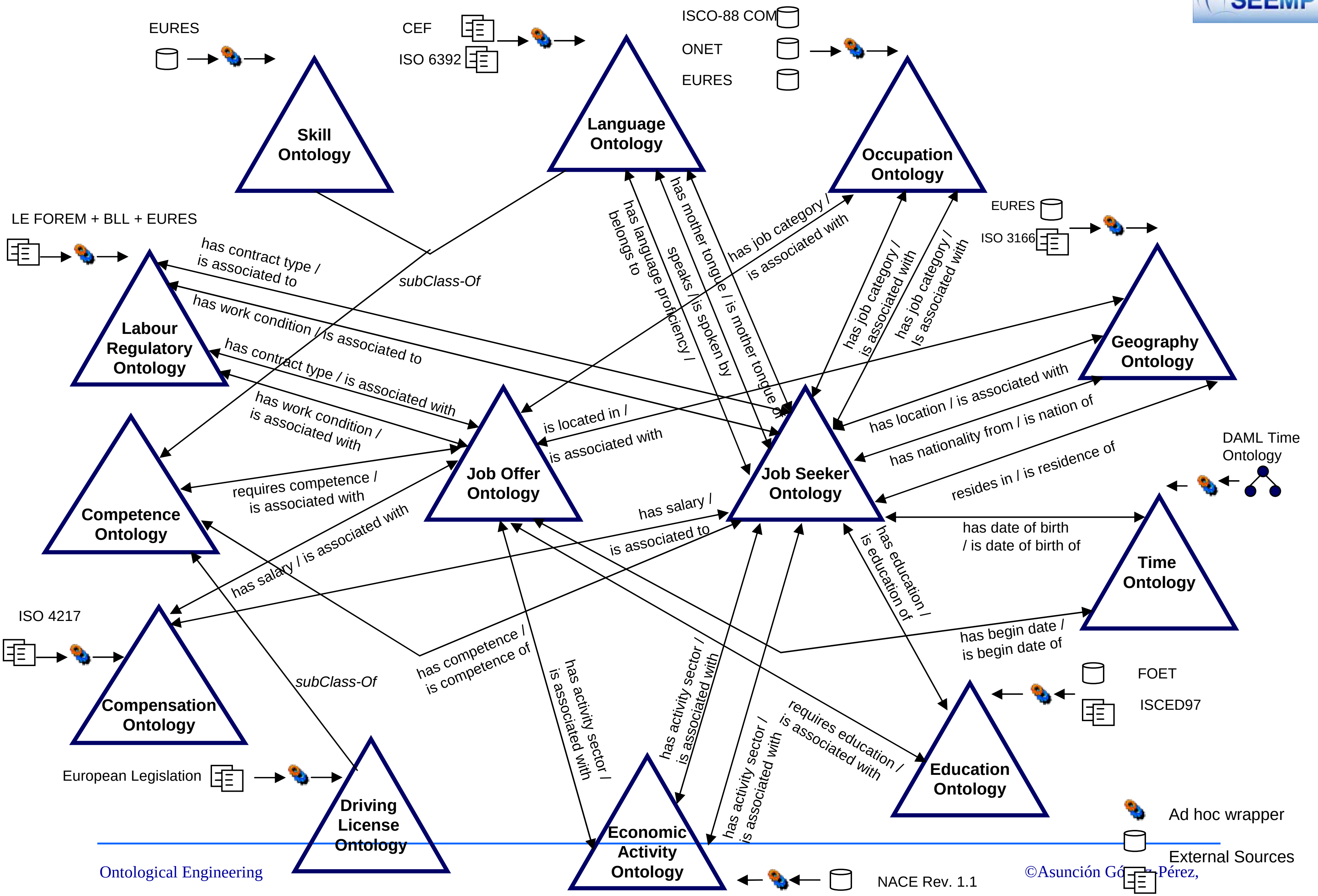
Conceptualization



ISCO-88 (COM), ONET, EURES taxonomy, FOET, ISCED97, NACE, ISO 4217, ISO 3166, ISO 6392, HR-XML, …

Conceptualization:
Modular approach for ontology construction


Reusability
–
+
Usability
+
–
Application Domain O. : *Job Seeker, Job Offer*
Domain O.: *Economic Activity, Occupation, Education, Skill, Driving License, Compensation, Labour Regulatory, Competence*
General/Common Ontologies: *Time, Geography, Language*
Representation Ontology: *WSML*
w l s m



EURES

CEF

ISO 6392

ISCO-88 COM

ONET

EURES

Skill Ontology

Language Ontology

Occupation Ontology

LE FOREM + BLL + EURES

EURES

ISO 3166

Labour Regulatory Ontology

Geography Ontology

Competence Ontology

Job Offer Ontology

Job Seeker Ontology

DAML Time Ontology

Time Ontology

ISO 4217

Compensation Ontology

Driving License Ontology

Economic Activity Ontology

Education Ontology

European Legislation

FOET

ISCED97

NACE Rev. 1.1

subClass-Of

subClass-Of

has contract type / is associated to

has work condition / is associated to

has contract type / is associated with

has work condition / is associated with

requires competence / is associated with

has salary / is associated with

has salary / is associated to

has competence / is competence of

has activity sector / is associated with

has activity sector / is associated with

has activity sector / is associated with

requires education / is associated with

has language proficiency / belongs to

speaks / is spoken by

has mother tongue / is mother tongue of

has job category / is associated with

has job category / is associated with

has job category / Is associated with

is located in / is associated with

has location / is associated with

has nationality from / is nation of

resides in / is residence of

has date of birth / is date of birth of

has education / is education of

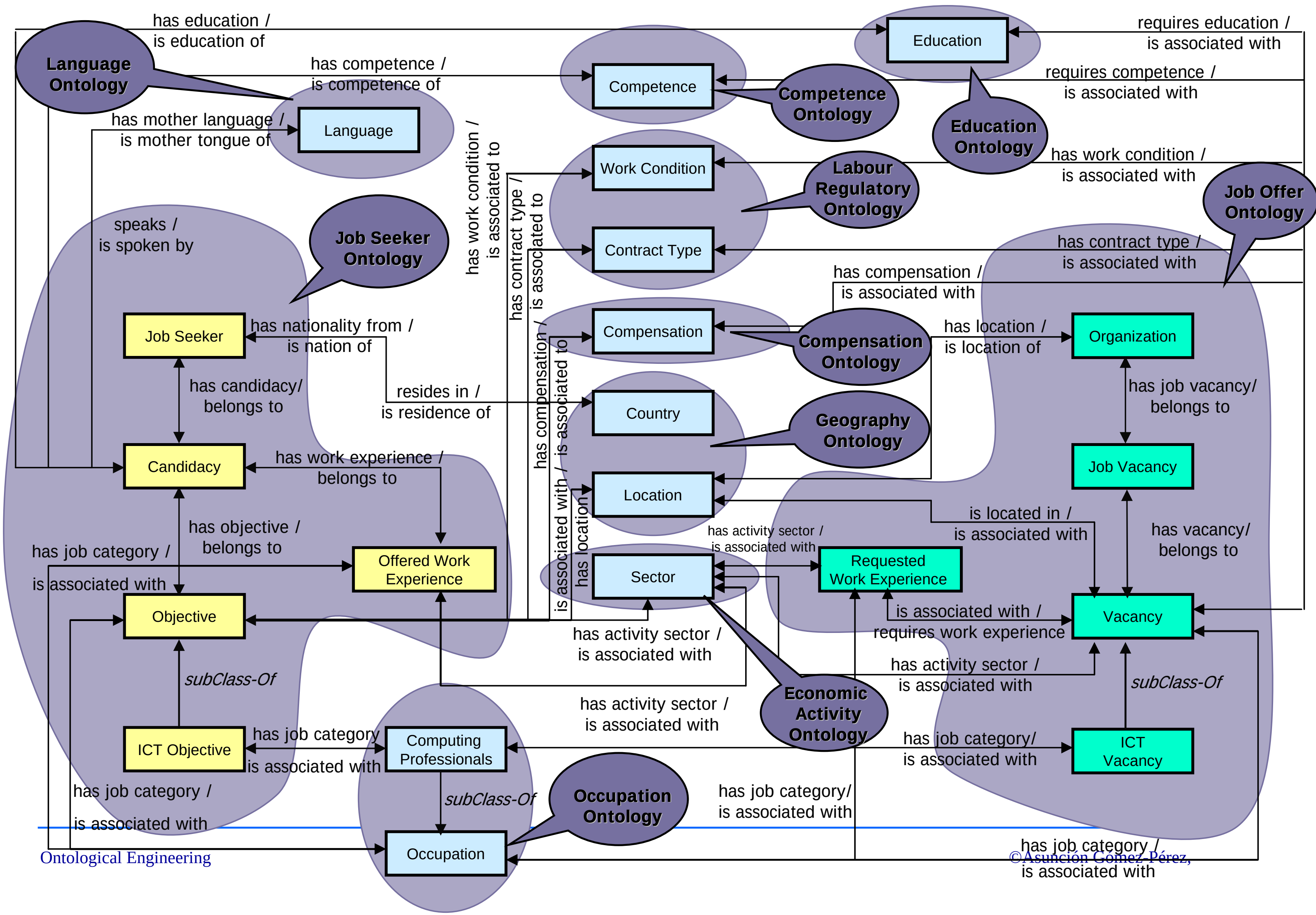
has begin date / is begin date of

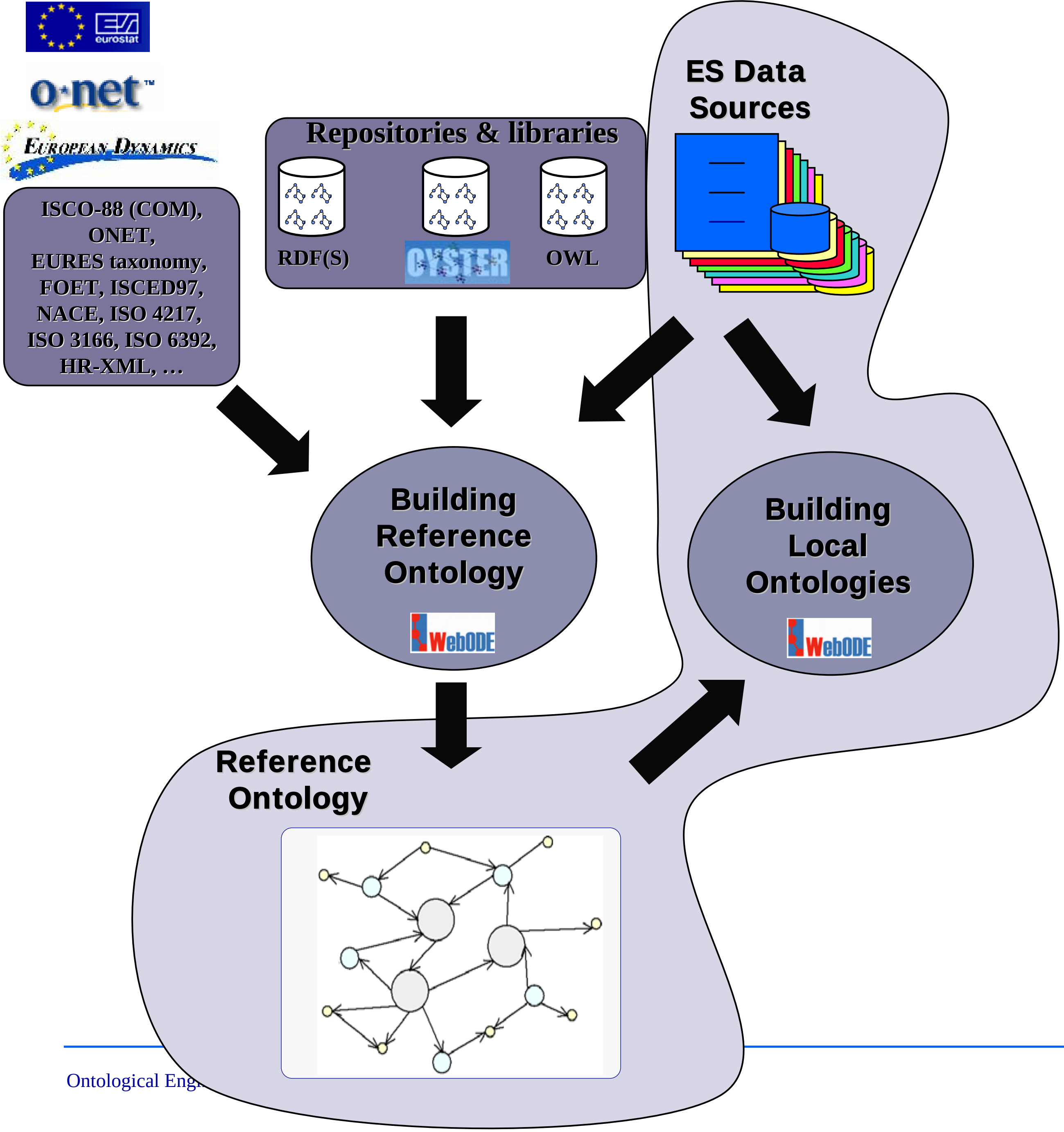
Ad hoc wrapper

External Sources

Details of the ontology

SEEMP


Language Ontology
Competence Ontology
Education Ontology
Labour Regulatory Ontology
Job Offer Ontology
Job Seeker Ontology
Compensation Ontology
Geography Ontology
Economic Activity Ontology
Occupation Ontology
Education
Competence
Language
Work Condition
Contract Type
Compensation
Country
Location
Sector
Job Seeker
Candidacy
Offered Work Experience
Objective
ICT Objective
Computing Professionals
Occupation
Organization
Job Vacancy
Requested Work Experience
Vacancy
ICT Vacancy
has education / is education of
has competence / is competence of
has mother language / is mother tongue of
speaks / is spoken by
has nationality from / is nation of
resides in / is residence of
has candidacy/ belongs to
has work experience / belongs to
has objective / belongs to
has job category / is associated with
subClass-Of
has job category / is associated with
has job category / is associated with
has work condition / is associated to
has contract type / is associated to
has compensation / is associated to
is associated with / has location
has activity sector / is associated with
has activity sector / is associated with
subClass-Of
requires education / is associated with
requires competence / is associated with
has work condition / is associated with
has contract type / is associated with
has compensation / is associated with
has location / is location of
has job vacancy/ belongs to
is located in / is associated with
has activity sector / is associated with
has vacancy/ belongs to
is associated with / requires work experience
has activity sector / is associated with
subClass-Of
has job category/ is associated with
has job category/ is associated with
has job category / is associated with



ISCO-88 (COM), ONET, EURES taxonomy, FOET, ISCED97, NACE, ISO 4217, ISO 3166, ISO 6392, HR-XML, …

Repositories & libraries

RDF(S)

OYSTER

OWL

ES Data Sources

Building Reference Ontology

WebODE

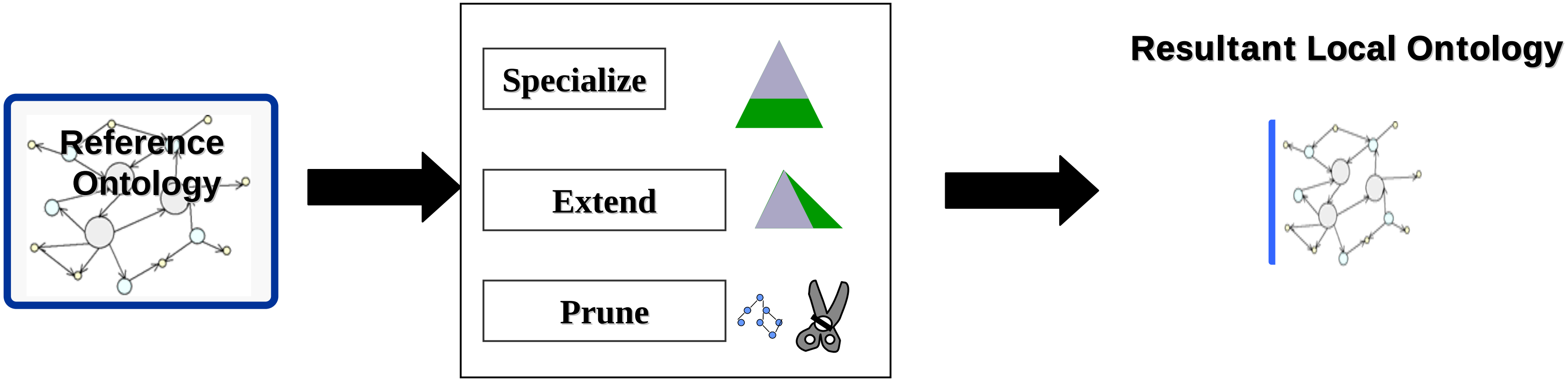
Building Local Ontologies

WebODE

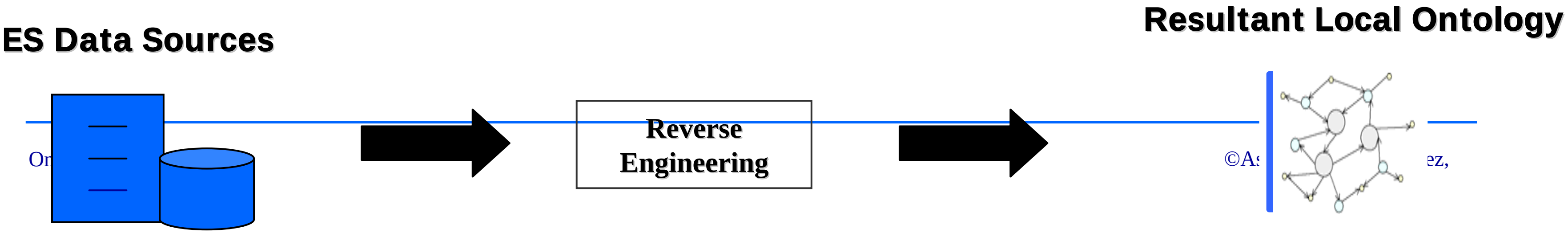
Reference Ontology

Local Ontologies Building Process

- Option 1: *Building Local Ontologies from the Reference Ontology.*



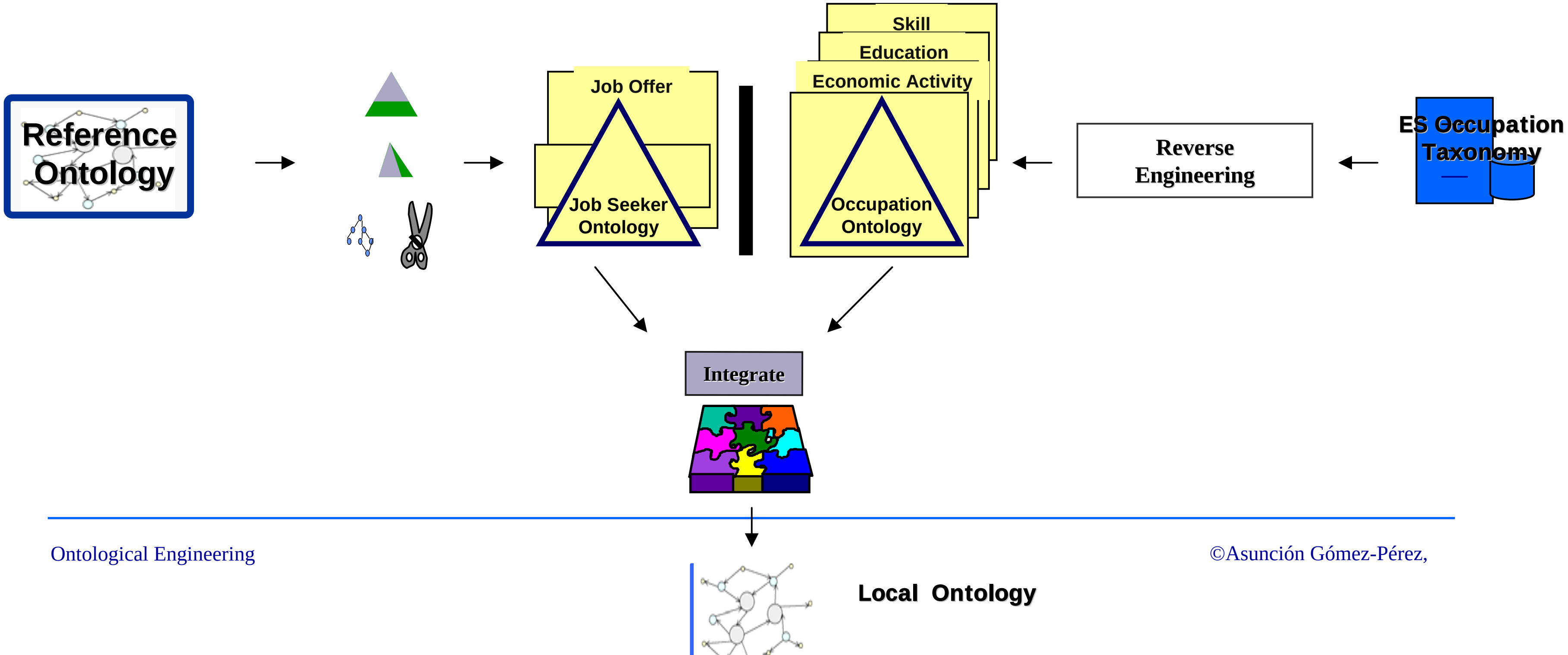
- Option 2: *Building Local Ontologies as a reverse engineering process from ES Data Sources.*



Hybrid approach for building Local Ontologies

A hybrid approach

- Option 1 for Job Seeker and Job Offer Ontologies
- Option 2 for Occupation, Education, etc.


Reference Ontology
Job Offer
Job Seeker Ontology
Skill
Education
Economic Activity
Occupation Ontology
Reverse Engineering
ES Occupation Taxonomy
Integrate
Local Ontology