Index

- ❑ Introduction

- ❑ Scenarios in Ontology Building

- ❑ Methodological Guidelines for Ontology Specification

- ❑ Quick Search of Existing Knowledge Resources

- ❑ Guidelines for Ontology Development Project Scheduling

- ❑ Methodological Guidelines for Non Ontological Resource Reuse and Reengineering

- ❑ Methodological Guidelines for Ontology Reuse

- ❑ **Methodological Guidelines for ODPs Reuse**

- ❑ Creating the Final Ontology Model

Ontology Engineer ingGroup

Patterns

Pattern is something proposed for imitation.

Design Pattern refers to shared guidelines that help solve design problems.

Ontology Design Pattern (ODP) is a modeling solution to solve a recurrent ontology design problem.

- Pattern is associable with the wider "**good/best practice**" of software engineering. It includes a wider range of solution types. For example: naming conventions in software engineering are considered good practices, they are not design patterns.

- ODPs can be classified into **six families**

  - Each family addresses different kinds of problems, and can be represented with different levels of formality.

Ontology Design Patterns

The idea of **applying patterns for modelling** ontologies was proposed by [Clark et al., 2000]

The **goal of the ODPs reuse** is
- to facilitate the solution of modelling issues
- to improve interoperability through using well-proven solutions and best practices, in the form of patterns

Catalogues

Clark, P., Thompson, J., & Porter, B. W. *Knowledge Patterns*. In KR2000: Principles of Knowledge Representation and Reasoning. pp. 591-600. 2000

Ontology Engineering Group

Types of Ontology Design Patterns
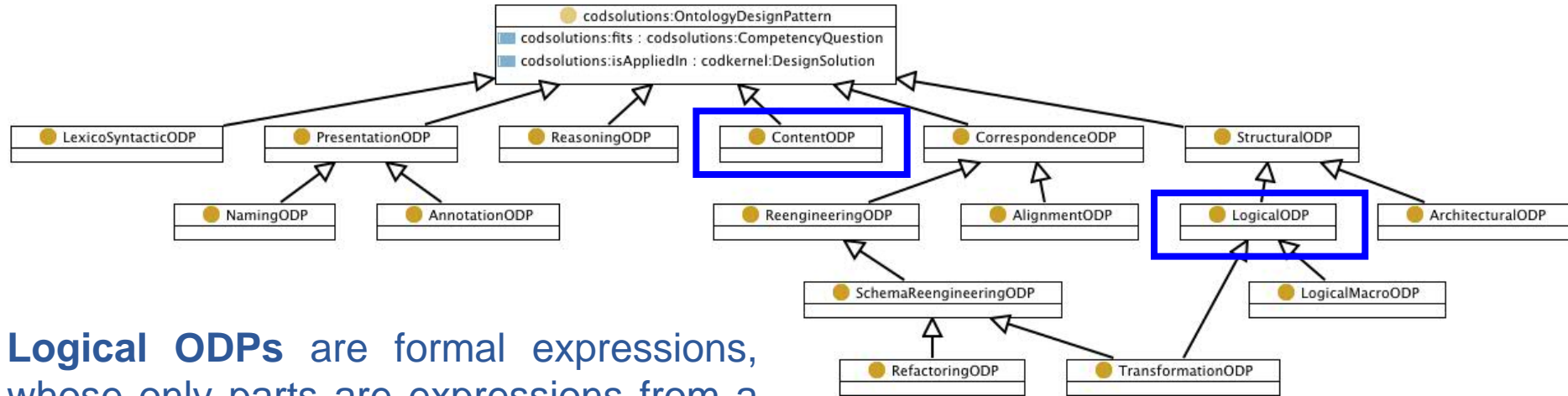
Logical ODPs are formal expressions, whose only parts are expressions from a logical vocabulary e.g., OWL DL, that solve a problem of expressivity

- Logical ODPs are independent from a specific domain of interest, i.e. they are content-independent
- Logical ODPs solve design problems where the primitives of the representation language do not directly support certain logical constructs

Content OPs (CPs) encode conceptual, rather than logical design patterns.

- Logical OPs solve design problems independently of a particular conceptualization.
- CPs propose patterns for solving design problems for the domain classes and properties that populate an ontology. They address content problems.

http://ontologydesignpatterns.org/wiki/OPTypes

Types of Ontology Design Patterns
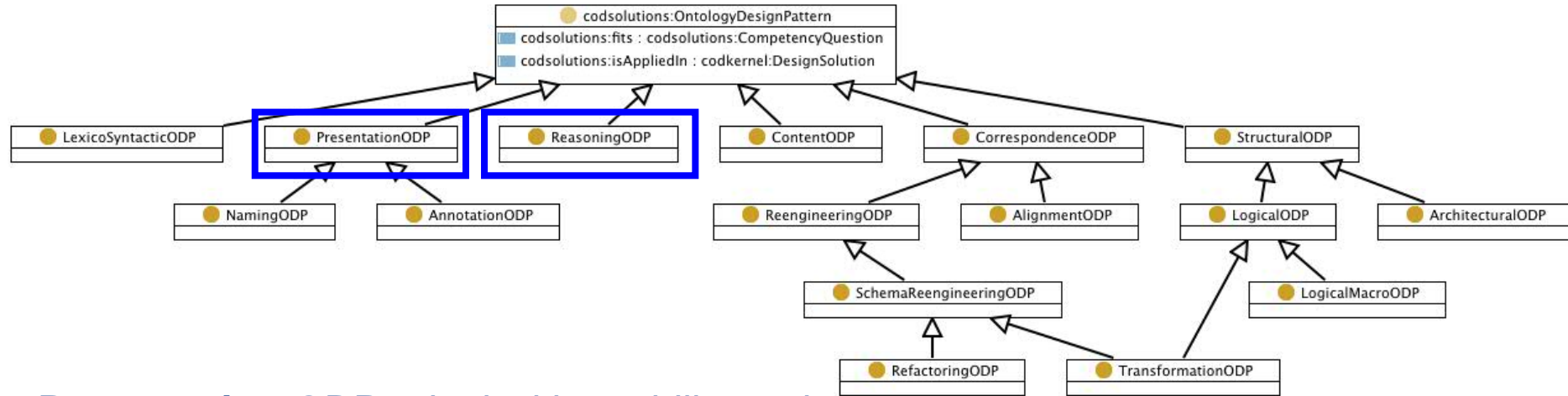
Presentation ODPs deal with usability and readability of ontologies from a user perspective

Reasoning ODPs are applications of Logical ODPs oriented to obtain certain reasoning results, based on the behavior implemented in a reasoning engine

http://ontologydesignpatterns.org/wiki/OPTypes

Inventory of Patterns

http://ontologydesignpatterns.org

| community:listpatterns | discussion | view source | history |

Community:ListPatterns

These are lists for available ODP catalogues.

Submissions

This area aims at collecting Ontology Design Pattern proposals from ODP users.

After the author has finished the submission and asked for a review, the proposals are assigned to at least two members of the ODP Quality Committee, review.

Positive reviews can be accompanied with guidelines for fixing possible problems of the proposed Content OP.

Once such problems have been addressed, the proposed Content OP can be certified and published in the official catalogue.

See the submissions list:

- Content ODPs
- Reengineering ODPs
- Alignment ODPs
- Logical ODPs
- Architectural ODPs
- Lexico-Syntactic ODPs

navigation

- Main page
- List patterns
- Pattern types
- Modeling Issues
- Domains
- Training
- Events

contribute

- Submit a pattern
- Submit an exemplary ontology
- Post a modeling issue
- Review a pattern
- Feedback about the portal
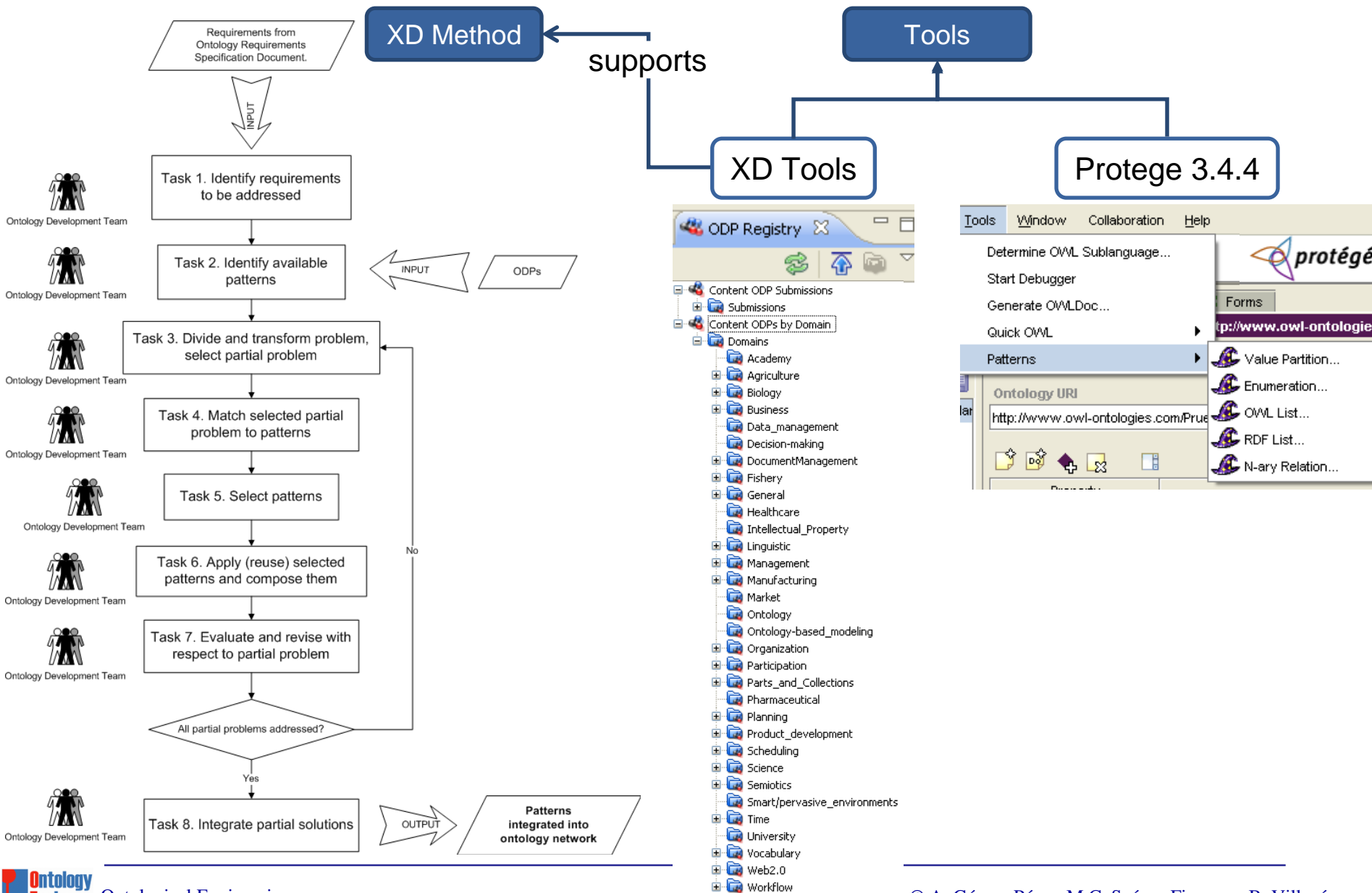- Request an ODP account

help

Ontology Design Patterns: how to reuse them

Requirements from Ontology Requirements Specification Document.

INPUT

Task 1. Identify requirements to be addressed
Ontology Development Team

Task 2. Identify available patterns
Ontology Development Team

INPUT ODPs

Task 3. Divide and transform problem, select partial problem
Ontology Development Team

Task 4. Match selected partial problem to patterns
Ontology Development Team

Task 5. Select patterns
Ontology Development Team

Task 6. Apply (reuse) selected patterns and compose them
Ontology Development Team

Task 7. Evaluate and revise with respect to partial problem
Ontology Development Team

All partial problems addressed?

No

Yes

Task 8. Integrate partial solutions
Ontology Development Team

OUTPUT

Patterns integrated into ontology network

XD Method

supports

Tools

XD Tools

Protege 3.4.4

ODP Registry

Content ODP Submissions
- Submissions
Content ODPs by Domain
- Domains
  - Academy
  - Agriculture
  - Biology
  - Business
  - Data_management
  - Decision-making
  - DocumentManagement
  - Fishery
  - General
  - Healthcare
  - Intellectual_Property
  - Linguistic
  - Management
  - Manufacturing
  - Market
  - Ontology
  - Ontology-based_modeling
  - Organization
  - Participation
  - Parts_and_Collections
  - Pharmaceutical
  - Planning
  - Product_development
  - Scheduling
  - Science
  - Semiotics
  - Smart/pervasive_environments
  - Time
  - University
  - Vocabulary
  - Web2.0
  - Workflow

Tools Window Collaboration Help

protégé

Determine OWL Sublanguage...
Start Debugger
Generate OWLDoc...
Quick OWL ▸
Patterns ▸

Forms
tp://www.owl-ontologies

Value Partition...
Enumeration...
OWL List...
RDF List...
N-ary Relation...

Ontology URI
http://www.owl-ontologies.com/Prue

Ontology Engineer ingGroup

Content Pattern Reuse: XD Method