Linked Data Workflow Project Ontology
- LDWPO v. 1.0 -

Ontology Development Process

Technical Report

(Document Version 0.1)

November 28, 2015

srautenberg@unicentro.br

Contents

1 Introduction

This report presents the ontology development process applied to establish the *Linked Data Workflow Project ontology* (LDWPO). This process is the result of the combination of some methodological artifacts and best practices from On-to-Knowledge [6], METHONTOLOGY [3], and Ontology Development 101 Guide [4]. In a nutshell, this combination is based on:

- **On-to-Knowledge** contributes to the specification activity, by applying competence questions to confirm the purpose and scope of an ontology;

- **Ontology Development 101** defines an interactive process in an ontology development; and

- **METHONTOLOGY** suggests a set of artifacts to document and evaluate the ontology development process.

Considering the listed methodologies, we define and use a process based on four main activities: i) *ontology specification*; ii) *knowledge acquisition*; iii) *ontology implementation*; iv) and *ontology verification*. The Figure 1 represents this process, relating all inner tasks for each activity.

In the next sections, we discuss briefly each of those activities, presenting the artifacts used for documenting the LDWPO development.

ONTOLOGY DEVELOPMENT PROCESS

Ontology Specification

- scope identification
- purpose identification
- knowledge source identification
- reuse consideration

Knowledge Acquisition

- competence questions generation
- searching terms in knowledge sources
- terms listing
- terms classification and definition
- hierarchy definition
- restrictions mapping
- object properties mapping
- data properties mapping
- object properties refinement
- data properties refinement

Ontology Implementation

- instances creation
- data properties allocation
- object properties allocation

Ontology Evaluation

- knowledge sources consideration
- initial framework consideration
- use cases consideration

Figure 1: Ontology development process applied to develop LDWPO.

2 Ontology Specification

This activity defines ontology development costs, taking into account:

- **The purpose identification** underlies why ontology has to be developed in regard to the existing ontology space.

- **The scope identification** answers the general questions as "who are the users?", "what are the intention for using it?", and so on.

- **The knowledge source identification** enumerates books, papers, dictionaries, and other sources with relevant information.

- **The reuse consideration** selects existing ontologies and vocabularies as candidates to provide established pieces of knowledge.

2.1 Identifying the purpose

The purpose of LDWPO addresses some provenance and reproducibility issues in a *Linked Data Workflow Project* (LDWProject). It models simple piece of knowledge about *project*s, *method*s, *plan*s, *execution*s, *artifact*s, and *people*, to describe the producing of RDF datasets in a systematic way. In a management point of view, this supports the maintainability of RDF dataset over time, considering the *Linked Data Lifecycle* [2].

2.2 Identifying the scope

1. Managing the lifecycle of RDF dataset maintaining;

2. Generating useful documents for reproducing the RDF dataset maintaining workflows; and

3. Mediating the use of *Linked Data Stack* technologies [1] for (semi-)automatized execution of workflows.

2.3 Identifying the knowledge sources

1. *Dublin Core* (**DC**)[1] - is a simple and powerful vocabulary used to describe web resources, promoting interoperability in Linked Data environments.

2. *Description of a Project* (**DOAP**)[2] - is a project to create an XML/RDF vocabulary to describe software projects, and in particular open source. Its use could be justify by contributing with a common vocabulary for projects.

3. *Friend of a Friend* (**FOAF**)[3] - is a project devoted to linking people and information using the Web. Its definitions could be used to link people to its actions, according the roles contributor or creator, in an LDWProject.

4. *Open Provenance Model Vocabulary* (**OPMV**)[4] - is a lightweight provenance vocabulary to assist the interoperability between provenance information on the Semantic Web. Using together with other vocabularies, such as DC and FOAF, it could provide resources for publishing.

5. *The PROV Ontology* (**PROV-O**)[5] - is used to represent and interchange provenance information generated in different systems and under different contexts. It could be used to share useful definitions.

6. *Publishing Workflow Ontology* (**PWO**)[6] - is an ontology for describing the workflow associated with the publication of a document. Also, it could be used to share useful definitions.

7. *Agile Knowledge Engineering and Semantic Web Research Group* (**AKSW**)[7] - is a group compromised to research/develop methods and tools for Linked Data in an engineering fashion. The site of this group can provide useful definitions to ontology resources, specially, to create the ontology instances.

8. *DBpedia*[8] - is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. Its knowledge base can be used for searching some ontology definitions.

---

[1] http://purl.org/dc/terms/
[2] https://github.com/edumbill/doap
[3] http://xmlns.com/foaf/0.1/
[4] http://purl.org/net/opmv/ns
[5] http://www.w3.org/ns/prov#
[6] http://purl.org/spar/pwo
[7] http://aksw.org/About.html
[8] http://dbpedia.org/About

2.4 Considering the reuse

We consider reusing resources from all the knowledge sources listed in subsection 2.3.

3 Knowledge Acquisition

This activity comprehends the conceptualization and formalization stages in an ontology development process. Knowledge acquisition could be seen as the point of major interaction among the knowledge engineers and domain experts. Here, most of ontology's elements are listed and defined. Interactively, it considers the following tasks:

- **Competence questions generation** includes interviewing the domain experts, asking them to elaborate the questions that the ontology must answer. It is an easy way to get/understand the terms, relationships used in a particular domain.

- **The terms listing** takes into account the competence questions, enumerating the common vocabulary used by the domain experts. Commonly, two lists are formed: *substantives*, for identify classes and data properties; and *verbs* for discovering inner object properties.

- **Searching terms in knowledge sources** considers the knowledge sources, enumerating common terms from the domain that did not appear in the set of competence questions.

- **The terms classification and definition** arranges the consensual knowledge of the domain, classifying the terms as classes, object properties, data properties, instances, or restrictions. In addition, it is needed to define the terms in natural language, resulting on an established consensus about the meanings.

- **The hierarchy definition** organizes the classes as a tree once the terms are classified (in a similar fashion as in Object Oriented Programming). Also takes care on inheritance of the data and object properties.

- **The restrictions mapping** verifies the rules that can restrict the content of data or object properties for each class.

- **The object properties mapping** assembles, for each class, the terms considered object properties that explicitly show the relationship from a particular class to another class(es) of the domain.

- **The data properties mapping** aggregates, for each class, the terms considered data properties that explicitly are features of the class.

- **The object properties refinement** sets up the functionality, transitivity, symmetrical, and reflexive features for each object property.

- **The data properties refinement** defines the supported datatype and its functionality feature for each data property.

3.1 Generating the competence questions

Below, we list the competence questions of the LDWPO, organized according the Zachmann Framework [5].

3.1.1 Dimension What

1. What are the people's names that contribute in this project?

2. What is the person's name that create this project?

3. What is the project's name?

4. What are the employed tools in this project?

5. What are the input datasets of this project?

6. What are the datasets of this project?

7. What are the stages of this project?

8. What are the transformations in this project?

9. What is the output dataset of this stage?

10. What are the steps done in this stage?

11. What is the previous step of this step?

12. What is the next step of this step?

13. What is the format of the dataset?

14. What are the project's goals?

15. What is the license of the dataset?

16. What is the kind of task of this step?

17. what are the tasks that this tool is applied?

3.1.2 Dimension Where

18. Where is the input dataset stored?

19. Where is the output dataset stored?

3.1.3 Dimension When

20. When was the step executed?

21. When did the workflow start?

22. When did the workflow finish?

3.1.4 Dimension Why

- *No questions were made.*

3.1.5 Dimension Who

23. Who are the people that contribute in this project?

24. Who is responsible for this step?

3.1.6 Dimension How

25. How is the input dataset stored?

26. How is the output dataset stored?

3.2 Listing terms from competence questions

- **Substantives** - dataset, format, input_dataset, license, next_step, output_dataset, person (people), person_name, previous_step, project, project_name, project_goal, responsible (contributor), stage, step, task, tool (employed tool), transformation, and workflow.

- **Verbs (in infinitive form)** - apply, be, contribute, create, do, employ, execute, finish, start, and store.

3.3 Enumerating the potential resources coming from knowledge sources

Knowledge source	Terms
DC	**Classes -** Agent, AgentClass, Dataset, Event, ProvenanceStatement, and Software. **Properties -** abstract, contributor, created, creator, date, description, format, hasVersion, identifier, provenance, publisher, rights, and title.
DOAP	**Classes -** Project, Version, Specification, and Repository. **Properties -** name, homepage, created, shortdesc, description, maintainer, developer, documenter, tester, helper, and audience.
FOAF	**Classes -** Agent, Document, Group, Organization, Person, and Project. **Properties -** homepage, maker, mbox, and name.
OPMV	**Classes -** Agent, Artifact, and Process. **Properties -** used, wasControlledBy, wasDerivedFrom, wasEncodedBy, wasGeneratedBy, wasPerformedBy, wasTriggeredBy, and wasUsedAt.
PROV-O	**Classes -** Activity, Agent, Creator, Derivation, Organization, Person, Plan/plan, Publisher, and SoftwareAgent.
PWO	**Classes -** Step, and Workflow. **Properties -** hasFirstStep, hasNextStep, hasPreviousStep, hasStep, involvesEvent, isInvolvedInStep, isNeededBy, isProducedBy, isStepOf, needs, and produces.
AKSW and/or LINKED DATA STACK	**Activity's instances -** Extraction, Storage/Querying, Manual Revision/Authoring, Interlink/Fusing, Classification/Enrichment, Quality analysis, Evolution/Repair, and Searching/Browsing/Exploration. **Tool's instances -** Virtuoso Sponger, DBpedia Spotlight, DBpedia Spotlight UI, pool party extractor, D2R, R2R, Apache Stanbol, CSVImport, Virtuoso 7 RDF Store, SparQLed, sparqlify, SIREn, OntoWiki, RDFAuthor, poolparty, LIMES, Silk, Sieve, DL-Learner, ORE, LODrefine, Sigma, Spatial Semantic Browser, CubeViz, and Facete.

Table 1: List of potential terms obtained from another knowledge sources.

3.4 Defining classes, data and object properties

3.4.1 Defining classes

1. **Artifact**. Abstract class that provides the `name` to the elements used to establish an `LDWStep`.
   rdf:label - Artifact.
   rdfs:seeAlso - [opmv:Artifact, and prov:Entity].

2. **Dataset**. Subclass of `Artifact` that represents a collection of data. Related to a `LDWStep`, it is used as an input or an output `Artifact`.
   rdf:label - Dataset.
   rdfs:seeAlso - [dc:Dataset].

3. **RDFDataset**. Subclass of `Dataset` that represents a collection of data adherent to a valid RDF format.
   rdf:label - RDFDataset.

4. **Homepage**. Subclass of `Artifact` that represents the the or main web page of an `LDWProject`.
   rdf:label - Homepage.

5. **Report**. Subclass of `Artifact` that represents a document generated by aggregating the `Messages` and `Statuses` of an `LDWorkflowExecution`.
   rdf:label - Report.

6. **Tool**. Subclass of `Artifact` that represents a computational implementation of one or more algorithm(s) suitable for processing data.
   rdf:label - Tool.
   rdfs:seeAlso - [dc:Software, prov:SoftwareAgent, foaf:Agent].

7. **ToolConfiguration**. Subclass of `Artifact` that represents a the setup file to be used by a `Tool`, when defining a `LDWStep`.
   rdf:label - ToolConfiguration.

8. **Format**. Class that represents the encode format of a `Dataset`, such as SQL, CSV, or RDF.
   rdf:label - Format.
   rdfs:seeAlso - [dc:MediaTypeOrExtent].

9. **KnowledgeBody**. Top abstract class for representing pieces of knowledge used on modeling an `LDWProject`, providing the `name` and `description` data properties.
   rdf:label - KnowledgeBody.

10. **Condition**. Subclass of `KnowledgeBody` that represents (i) the obligations to be cover before an `LDWorkflowExecution` is performed (object property: precondition), and (ii) the settings achieved after the `LDWorkflowExecution` is successfully finished (object property: postcondition).
    rdf:label - Condition.

11. **Execution**. Subclass of `KnowledgeBody` and abstract class for `LDWorkflow-Execution` and `LDWStepExecution` used for describing the execution issues an `LDWProject`. It describes a particular point of time with `startedDate` and `endedDate` data properties.
    rdf:label - Execution.

12. **LDWorkflowExecution**. Subclass of `Execution` related to a particular execution of an `LDWorkflow`. Inside, it organizes a linked list of `LDWStepExecutions`.
    rdf:label - LDWorkflowExecution.

13. **LDWStepExecution**. Subclass of `Execution` and `Step` that represents an atomic unit of `LDWorkflowExecution`. It is related to a particular `LDWStep`.
    rdf:label - LDWStepExecution.

14. **LDWProject**. Subclass of `KnowledgeBody` that represents an endeavour of creating or maintaining `RDFDatasets`.
    rdf:label - LDWProject.
    rdfs:seeAlso - [foaf:Project].

15. **Method**. Subclass of `KnowledgeBody` and abstract class for `Process`, `Activity`, and `Task` Used to describe the methodological aspects of an `LDWProject`.
    rdf:label - Method.

16. **Process**. Subclass of `Method` that represents a concise set of `Activities` used to maintain an `RDFDataset`.
    rdf:label - Process.

17. **Activity**. Subclass of `Method` that represents a concise set of `Tasks` used to maintain an `RDFDataset`.
    rdf:label - Activity.

18. **Task**. Subclass of `Method` that represents the atomic unit of an established `Process`. It describes a best-practice for maintaining an `RDFDataset`.
    rdf:label - Task.

19. **Plan**. Subclass of `KnowledgeBody` and abstract class for `LDWorkflow` and `LDWStep` used for describing the planning issues an `LDWProject`.
    rdf:label - Plan.

20. **LDWorkflow**. Subclass of `LDWorkflow` related to describe a particular planning activity of an `LDWProject` for maintaining an `RDFDataset`. Inside, it organizes a linked list of `LDWSteps`.
    rdf:label - LDWorkflow.
    rdfs:seeAlso - [pwo:Workflow].

21. **LDWStep**. Subclass of `Plan` and `Step` that represents an atomic unit of `LDWorkflow`.
    rdf:label - LDWStep.
    rdfs:seeAlso - [pwo:Step, opmv:Process, and prov:Activity].

22. **License**. Class that represents the type of access permission to a `Dataset`.
    rdf:label - License.
    rdfs:seeAlso - [dc:LicenseDocument].

23. **Location**. Class that represents the path to access an `Artifact`.
    rdf:label - Location.
    rdfs:seeAlso - [prov:Location].

24. **Message**. Class that represents the feedback report originating from applying a `Tool` and a `ToolConfiguration` in order to modify an `Dataset` during an `LDWStepExecution`.
    rdf:label - Message.

25. **Person**. Class that represents people involved in an `LDWProject`, performing the roles of contributor (for `Steps`) or creator (for whole `LDWProject`).
    rdf:label - Person.
    rdfs:seeAlso - [opmv:Agent, prov:Agent, prov:Person, dc:Agent, foaf:Agent, and foaf:Person].

26. **Status**. Class that represents the classification for a `Message` (successful finished, unsuccessful finished, aborted, etc.).
    rdf:label - Status.

27. **Step**. Abstract class for `LDWStep` and `LDWStepExecution` used to describe the previous and next step for a given `Step`. It is used to define the linked list of the `LDWorkflow` and `LDWorkflowExecution`. It is related to a particular contributor (`Person`).
    rdf:label - Step.

3.4.2 Defining object properties

1. **activity**. Object property that points to the `activitys` of a `Process`. As restrictions, one `Process` has one or more `Activities` and one `Activity` is related to one or more `Processes`.
   rdf:label - activity.

2. **contributor**. Object property that relates a `Person` to the `Steps`. As restrictions, one `Step` can be performed by one `Person` and one `Person` can be related to one or more `Steps`.
   rdf:label - contributor.
   rdfs:seeAlso - [opmv:wasControlledBy, opmv:wasPerformedBy, prov:wasAssociatedWith, dc:contributor].

3. **creator**. Object property that points to a `Person` who created an `LDWProject`. As restrictions, one `LDWProject` is created by one or more `Person` and one `Person` can be related to one or more `LDWProjects`.
   rdf:label - creator.
   rdfs:seeAlso - [opmv:wasControlledBy, and dc:creator].

4. **firstLdwStep**. Object property that points to the initial `LDWStep` of an `LDWork-flow`. As restrictions, one `LDWorkflow` has one `LDWStep` as the `firstLdw-Step` and one `LDWStep` can be the `firstLdwStep` in one or more `LDWork-flows`.
   rdf:label - firstLdwStep.
   rdfs:seeAlso - [pwo:hasFirstStep].

5. **firstLdwStepExecution** Object property that points to the initial `LDWStep-Execution` of an `LDWorkflowExecution`. As restrictions, one `LDWork-flowExecution` has one `LDWStepExecution` as `firstLdwStepExecu-tion` and one `LDWStepExecution` is the `firstLdwStepExecution` of a single `LDWorkflowExecution`.
   rdf:label - firstLdwStepExecution.

6. **format**. Object property that points to the `Format` of a `Dataset`. As restrictions, one `Format` is related to one or more `Dataset`s and a `Dataset` is related to a single `Format`.
   rdf:label - format.
   rdfs:seeAlso - [dc:format].

7. **homepage**. Object property that points to the `Homepage` of a `LDWProject`. As restrictions, one `Homepage` is related to one or more `LDWProject`s and a `LDWProject` is related to a single `Homepage`.
   rdf:label - homepage.
   rdfs:seeAlso - [foaf:homepage].

8. **inputDataset**. Object property that points to the `Dataset`s used as input by an `LDWStep`. As restrictions, one `LDWStep` has one or more `Dataset`s as input and one `Dataset` can be related as input to one or more `LDWStep`s.
   rdf:label - inputDataset.
   rdfs:seeAlso - [pwo:needs, opmv:used, and prov:used].

9. **ldWorkflow**. Object property that points to the `LDWorkflow`s related to an `LDWProject`. As restrictions, one `LDWProject` has one or more `LDWork-flows` and one `LDWorkflow` is related to a single `LDWProject`.
   rdf:label - ldWorkflow.

10. **ldWorkflowExecution**. Object property that points to the `LDWorkflowExecu-tions` related to an `LDWorklow`. As restrictions, one `LDWorkflow` has one or more `LDWorkflowExecutions` and one `LDWorkflowExecution` is re-lated to a single `LDWorkflow`.
    rdf:label - ldWorkflowExecution.

11. **ldwStep**. Object property that points to the `LDWSteps` related to an `LDWorklow`. As restrictions, one `LDWorkflow` has one or more `LDWSteps` and one `LDWStep` is related to one or more `LDWorkflows`.
    rdf:label - ldwStep.
    rdfs:seeAlso - [pwo:hasStep].

12. **ldwStepExecution**. Object property that points to the `LDWStepExecutions` related to an `LDWorklowExecution` and an `LDWStep`. As restrictions, one `LDWStepExecution` is related to a single `LDWStep` and to a single `LDWorkflowExecution`, and both `LDWorkflowExecution` and `LDWStep` are related to one or more `LDWStepExecution`.
    rdf:label - ldwStepExecution.

13. **license**. Object property that points to the type of `License` of a `Dataset`. As restrictions, one `Dataset` has one `License` and one `License` is related to one or more `Datasets`.
    rdf:label - license.
    rdfs:seeAlso - [dc:license].

14. **location**. Object property that points to the physical location of an `Artifact`, which could be an URL or local filesystem path. As restrictions, one `Artifact` has one `Location` and one `Location` is related to one or more `Artifacts`.
    rdf:label - location.
    rdfs:seeAlso - [prov:atLocation].

15. **message**. Object property that points to the `Message` generated as feedback when a `LDWStepExecution` is executed. As restrictions, one `LDWStepExecution` has one `Message` and one `Message` is related to one `LDWStepExecution`.
    rdf:label - message.

16. **nextStep**. Object property that points to the next `Step` into a linked list of `Steps`. It has `previousStep` as an inverse property. As restriction, one `Step` has only one next `Step` and is an irreflexive and asymmetric object property.
    rdf:label - nextStep.
    rdfs:seeAlso - [pwo:hasNextStep].

17. **outputDataset**. Object property that points to the `Datasets` to be transformed in an `LDWStep`. As restrictions, one `LDWStep` has one or more `Dataset` as output and one `Dataset` can be related as output to one or more `LDWSteps`.
    rdf:label - outputDataset.
    rdfs:seeAlso - [pwo:produces, and prov:generated].

18. **postcondition**. Object property that points to the `Condition` to achieve after an `LDWorkflow` is executed by an `LDWorkflowExecution`. As restrictions, one `LDWorkflow` has one or more `Conditions` as postcondition and one `Condition` can be related as postcondiction to one or more `LDWorkflows`.
    rdf:label - postcondition.

19. **precondiction**. Object property that points to the `Condition` to be presented before an `LDWorkflow` is executed by an `LDWorkflowExecution`. As restrictions, one `LDWorkflow` has one or more `Conditions` as precondition and one `Condition` can be related as precondition to one or more `LDWorkflows`.
    rdf:label - precondition.

20. **previousStep**. Object property that points to the previous `Step` into a linked list of `Steps`. It has `nextStep` as an inverse property. As restriction, one `Step` has only one previous `Step` and is an irreflexive and asymmetric object property.
rdf:label - previousStep.
rdfs:seeAlso - [pwo:hasPreviousStep, opmv:wasTriggeredBy, and
prov:wasInformedBy].

21. **report**. Object property that points to the `Report` generated as feedback when an `LDWorkflowExecution` is executed. It groups all `Messages` and `Statuses` from the related `LDWStepExecution`. As restriction, one `LDWorkflowExecution` has only one `Report` and one `Report` is related to only one `LDWorkflowExecution`.
rdf:label - report.

22. **status**. Object property that points to the `Status` of executing a `LDWStepExecution`. As restriction, one `LDWStepExecution` has only one `Status` and one `Status` is related to one or more `LDWStepExecution`.
rdf:label - status.

23. **task**. Object property that points to the `Task` of an `LDWStep` or `Activity`. As restrictions, one `Task` is related one or more `Activities` and one or more `LDWSteps`, and one `LDWStep` and one `Activity` are related to one or more `Tasks`.
rdf:label - task.

24. **tool**. Object property that points to the `Tool` employed in a `LDWStep`. As restrictions, one `Step` can employ one `Tool` and one `Tool` is related to one or more `LDWSteps`.
rdf:label - tool. rdfs:seeAlso - [pwo:needs, opmv:used, and prov:used].

25. **toolConfiguration**. Object property that points to the `ToolConfiguration` used by a `Tool` in a `LDWStep`. As restrictions, one `Tool` uses one or more `ToolConfigurations` and one `LDWStep` uses one `ToolConfigurations`. And one `ToolConfiguration` can be used by one `Tool` and also used by one or more `LDWStep`.
rdf:label - toolConfiguration.
rdfs:seeAlso - [pwo:needs, and opmv:used].

3.4.3 Defining data properties

1. **command**. Data property that contains the console command of a planned `LDWStep`. It relates the `Tool` and `ToolConfiguration` to be used by an `LDWStepExecution`. As restriction, `command` is a functional property and accepts only xsd:String values.
rdf:label - command.

2. **description**. Data property used by a `KnwoledgeBody` to describe instances. As restriction, `description` is a functional property and accepts only xsd:String

values.
rdf:label - description.
rdfs:seeAlso - [dc:description].

3. **endedDate**. Data property that expresses when an `Execution` instance is finished. As restriction, `endedDate` is a functional property and accepts only xsd:DateTime values.
   rdf:label - endedDate.
   rdfs:seeAlso - [dc:date, opmv:wasEndedAt, and prov:endedAtTime].

4. **goal**. Data property that expresses the set of goals of an `LDWProject`. As restriction, `goal` is a functional property and accepts only xsd:String values.
   rdf:label - goal.

5. **name**. Data property used by `Artifact`, `KnwoledgeBody`, `License`, and `Person`, to name instances. As restriction, `name` is a functional property and accepts only xsd:String values.
   rdfs:seeAlso - [dc:identifier, foaf:name].
   rdf:label - name.

6. **startedDate.** Data property that expresses when an `Execution` instance is started. As restriction, `startedDate` is a functional property and accepts only xsd:DateTime values.
   rdfs:seeAlso - [dc:date, dc:created, prov:startedAtTime,
   opmv:wasStartedAt].
   rdf:label - startedDate.

7. **value**. Data property used by `Format`, `KnwoledgeBody`, `Location`, `Message`, and `Status`, to attribute a content to the instances. As restriction, `value` is a functional property and accepts only xsd:String format.
   rdf:label - value.

3.5 Defining class hierarchy

The Figure 2 represents the class hierarchy to LDWPO - version 1.0
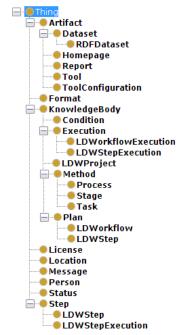

Figure 2: LDWPO class hierarchy.

3.6 Property Refinement and Restriction Mapping

The main purposes of Property Refinement is establishing the domain and the range for the object properties and data properties.

In parallel, the Restriction Mapping sets up the semantic characteristics of a particular property. For data properties, we define the supported data type and the functionality of the feature. For object properties, we define the functional, inverse functional, reflexive, irreflexive, symmetric, asymmetric, and transitive characteristics.

The achieved results of the Property Refinement and Restrictions Mapping tasks are shown in Table 2 and Table 3.

3.7 Mapping and refining object properties

Object Property	Domain	Range	Characteristics
contributor	Step	Person	[F]
creator	LDWProject	Person	[–]
format	Dataset	Format	[F]
firstLdwStep	LDWorkflow	LDWStep	[F]
firstLdwStepExecution	LDWorkflowExecution	LDWStepExecution	[F, IF]
homepage	LDWProject	Homepage	[F]
inputDataset	LDWStep	Dataset	[–]
ldWorkflow	LDWProject	LDWorkflow	[IF]
ldWorkflowExecution	LDWorkflow	LDWorkflowExecution	[IF]
ldwStep	LDWorkflow	LDWStep	[–]
ldwStepExecution	LDWorkflowExecution, LDWStep	LDWStepExecution	[IF]
license	Dataset	License	[F]
location	Artifact	Location	[F]
message	LDWStepExecution	Message	[F, IF]
nextStep	Step	Step	[I, A]
outputDataset	LDWStep	Dataset	[IF]
postcondiction	LDWorkflow	Condition	[–]
precondiction	LDWorkflow	Condition	[–]
previousStep	Step	Step	[I, A]
report	LDWorkflowExecution	Report	[F, IF]
activity	Process	Activity	[–]
status	LDWorkflowExecution	Status	[F]
task	Activity, LDWStep	Task	[–]
tool	LDWStep	Tool	[F]
toolConfiguration	LDWStep, Tool	ToolConfiguration	[–]

Table 2: Object properties - domains, characteristics, and ranges. For Characteristics: (F) - Functional, (IF) - Inverse Functional, (R) - Reflexive, (I) - Irreflexive, (S) - Symmetric, (A) - Asymmetric, and (T) - Transitive, (–) - no characteristic define
.

3.8 Mapping and refining data properties

Data Property	Domain	Range	Functional
command	LDWStep	String	Yes
description	KnowledgeBody	String	Yes
endedDate	Execution	DateTime	Yes
goal	LDWProject	String	Yes
name	KnowledgeBody, Artifact, License, Person	String	Yes
startedDate	Execution	DateTime	Yes
value	Message, Status, Format, Location	String	Yes

Table 3: Data properties - domains and ranges.

3.9 Ontology diagram

As the final result of the knowledge acquisition activity, a complete version of LDWPO is modeled. The Figure 3 presented below expresses the achieved result.
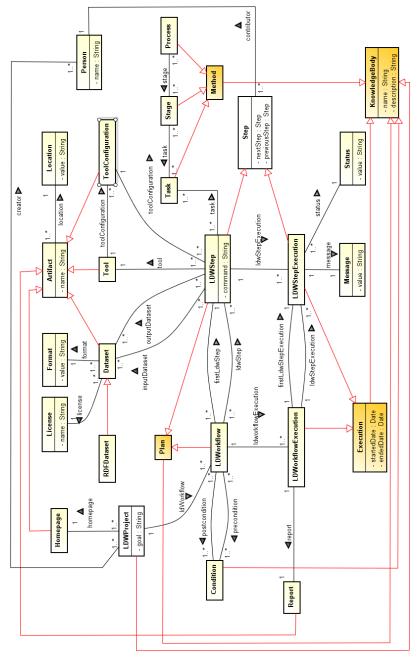

Figure 3: Representing LDWPO v. 1.0, highlighting the abstract concepts Knowledge Body, Method, Plan, and Execution.

4 Ontology Implementation

With minor interaction with domain experts, this activity is reserved to:

- **The instances creation** specifies the terms considered as samples of each class.

- **The data properties allocation** assigns the value(s) of internal properties for each instance.

- **The object properties allocation** connects a particular instance to the other instances of the domain, according the object properties.

As main result of this activity, we developed the first version of LDWPO. The ontology is available at `https://github.com/AKSW/ldwpo/blob/master/1.0/ldwpo.owl`.

5 Ontology Verification

Verification is related to check if an ontology meets the requirements by:

- **The knowledge sources consideration** compares the consistency of the represented knowledge in the ontology to the accepted understanding of the domain based on the knowledge sources. Basically, we relate the LDWPO resources to the resources of related knowledge sources, according a common understanding about the inner definitions. The relations are expressed at the level of rdfs:seeAlso, considering as knowledge sources: DC, FOAF, OPMV, PROV-O, and PWO.

- **The initial framework consideration** confirms the accuracy and completeness of the ontology, taking into account the purpose, scope, and mainly, by prototyping the competence questions as sparql queries.

- **Use cases consideration** inspects the usefulness of the ontology by simulating some scenarios.

5.1 Knowledge source consideration

In the next, we presented Table 4, Table 5, and Table 6 to expresses the consistency of LDWPO in terms of classes, object properties, and data properties, respectively.

5.1.1 Knowledge source consideration - Classes

Class	rdfs:seeAlso
Artifact	opmv:Artifact
Artifact	prov:Entity
Dataset	dc:Dataset
Tool	dc:Software
Tool	prov:SoftwareAgent
Tool	foaf:Agent
Format	dc:MediaTypeOrExtent
LDWProject	foaf:Project
LDWorkflow	pwo:Workflow
LDWStep	pwo:Step
LDWStep	opmv:Process
LDWStep	prov:Activity
License	dc:LicenseDocument
Location	prov:Location
Person	opmv:Agent
Person	prov:Agent
Person	prov:Person
Person	dc:Agent
Person	foaf:Agent
Person	foaf:Person

Table 4: List of classes and its equivalence to another knowledge sources [rdfs:seeAlso].

5.1.2 Knowledge source consideration - Object properties

Object property	rdfs:seeAlso
contributor	opmv:wasControlledBy
contributor	opmv:wasPerformedBy
contributor	prov:wasAssociatedWith
contributor	dc:contributor
creator	opmv:wasControlledBy
creator	dc:creator
format	dc:format
firstLdwStep	pwo:hasFirstStep
homepage	foaf:homepage
inputDataset	pwo:needs
inputDataset	opmv:used
inputDataset	prov:used
ldwStep	pwo:hasStep
license	dc:license
location	prov:atLocation
nextStep	pwo:hasNextStep
outputDataset	pwo:produces
outputDataset	prov:generated
previousStep	pwo:hasPreviousStep
previousStep	opmv:wasTriggeredBy
previousStep	prov:wasInformedBy
tool	pwo:needs
tool	opmv:used
tool	prov:used
toolConfiguration	pwo:needs
toolConfiguration	opmv:used

Table 5: List of object properties and its equivalence to another knowledge sources [rdfs:seeAlso].

5.1.3 Knowledge source consideration - Data properties

Data property	rdfs:seeAlso
description	dc:description
endedDate	opmv:wasEndedAt
endedDate	prov:endedAtTime
endedDate	dc:date
name	dc:identifier
name	foaf:name
startedDate	opmv:wasStartedAt
startedDate	prov:startedAtTime
startedDate	dc:date

Table 6: List of data properties and its equivalence to another knowledge sources [rdfs:seeAlso].

5.2 Initial framework consideration

Here, it is presented all competence questions listed in the knowledge acquisition activity that guided us to understand the purpose and scope of the LDWPO. These questions were used as requirements to be attended by the ontology approach. By developing the related sparql queries, it was possible to confirm the coverage of LDWPO.

5.2.1 What are the people's names that contribute in this project?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject ?nameContributor WHERE {
8    ?project            rdf:type                ldwpo:LDWProject   .
9    ?project            ldwpo:name              ?nameProject       .
10   ?project            ldwpo:ldWorkflow        ?ldworkflow        .
11   ?ldworkflow         ldwpo:firstLdwStep      ?step              .
12   ?ldworkflow         ldwpo:ldWorkflowExecution ?workflowExecution .
13   ?step               (ldwpo:nextStep)*       ?linkedStep        .
14   ?linkedStep         ldwpo:ldwStepExecution  ?stepExecution     .
15   ?workflowExecution  ldwpo:ldwStepExecution  ?stepExecution     .
16   ?stepExecution      ldwpo:contributor       ?contributor       .
17   ?contributor        ldwpo:name              ?nameContributor   .
18 } ORDER BY ?project
```

5.2.2 What is the person's name that create this project?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject ?nameCreator WHERE {
8    ?project  rdf:type       ldwpo:LDWProject .
9    ?project  ldwpo:name     ?nameProject     .
10   ?project  ldwpo:creator  ?creator         .
11   ?creator  ldwpo:name     ?nameCreator     .
12 } ORDER BY ?project
```

5.2.3 What is the project's name?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject WHERE {
8    ?project  rdf:type     ldwpo:LDWProject .
9    ?project  ldwpo:name   ?nameProject     .
10 } ORDER BY ?project
```

5.2.4 What are the employed tools in this project?

```
1  PREFIX rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:    <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:    <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:   <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo:  <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject ?nameTool WHERE {
8    ?project      rdf:type            ldwpo:LDWProject .
9    ?project      ldwpo:name          ?nameProject    .
10   ?project      ldwpo:ldWorkflow    ?ldworkflow     .
11   ?ldworkflow   ldwpo:firstLdwStep  ?step           .
12   ?step         (ldwpo:nextStep)*   ?linkedStep     .
13   ?linkedStep   ldwpo:tool          ?tool           .
14   ?tool         ldwpo:name          ?nameTool       .
15 } ORDER BY ?project
```

5.2.5 What are the input datasets of this project?

```
1  PREFIX rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:    <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:    <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:   <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo:  <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject ?nameDataset WHERE {
8    ?project       rdf:type            ldwpo:LDWProject  .
9    ?project       ldwpo:name          ?nameProject      .
10   ?project       ldwpo:ldWorkflow    ?ldworkflow       .
11   ?ldworkflow    ldwpo:firstLdwStep  ?step             .
12   ?step          (ldwpo:nextStep)*   ?linkedStep       .
13   ?linkedStep    ldwpo:inputDataset  ?inputDataset     .
14   ?inputDataset  ldwpo:name          ?nameDataset      .
15 } ORDER BY ?project
```

5.2.6 What are the datasets of this project?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?nameDataset WHERE {
8   {
9     SELECT ?nameProject ?nameDataset WHERE {
10      ?project        rdf:type            ldwpo:LDWProject  .
11      ?project        ldwpo:name          ?nameProject      .
12      ?project        ldwpo:ldWorkflow    ?ldworkflow       .
13      ?ldworkflow     ldwpo:firstLdwStep  ?step             .
14      ?step           (ldwpo:nextStep)*   ?linkedStep       .
15      ?linkedStep     ldwpo:inputDataset  ?inputDataset     .
16      ?inputDataset   ldwpo:name          ?nameDataset      .
17    }
18  }
19  UNION
20  {
21    SELECT ?nameProject ?nameDataset WHERE {
22      ?project        rdf:type            ldwpo:LDWProject  .
23      ?project        ldwpo:name          ?nameProject      .
24      ?project        ldwpo:ldWorkflow    ?ldworkflow       .
25      ?ldworkflow     ldwpo:firstLdwStep  ?step             .
26      ?step           (ldwpo:nextStep)*   ?linkedStep       .
27      ?linkedStep     ldwpo:outputDataset ?outputDataset    .
28      ?outputDataset  ldwpo:name          ?nameDataset      .
29    }
30  }
31 } ORDER BY ?project
```

5.2.7 What are the stages of this project?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?nameActivity WHERE {
8   ?project        rdf:type            ldwpo:LDWProject  .
9   ?project        ldwpo:name          ?nameProject      .
10  ?project        ldwpo:ldWorkflow    ?ldworkflow       .
11  ?ldworkflow     ldwpo:firstLdwStep  ?step             .
12  ?step           (ldwpo:nextStep)*   ?linkedStep       .
13  ?linkedStep     ldwpo:task          ?task             .
14  ?activity       rdf:type            ldwpo:Activity    .
15  ?activity       ldwpo:task          ?task             .
16  ?activity       ldwpo:name          ?nameActivity     .
17 } ORDER BY ?project
```

5.2.8 What are the transformations in this project?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?description WHERE {
8   ?project        rdf:type            ldwpo:LDWProject  .
9   ?project        ldwpo:name          ?nameProject      .
10  ?project        ldwpo:ldWorkflow    ?ldworkflow       .
11  ?ldworkflow     ldwpo:firstLdwStep  ?step             .
12  ?step           (ldwpo:nextStep)*   ?linkedStep       .
13  ?linkedStep     ldwpo:description   ?description      .
14  ?ldworkflow     ldwpo:ldwStep       ?linkedStep       .
15 }
```

5.2.9 What is the output dataset of this stage?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?nameDataset WHERE {
8   ?project        rdf:type            ldwpo:LDWProject  .
9   ?project        ldwpo:name          ?nameProject      .
10  ?project        ldwpo:ldWorkflow    ?ldworkflow       .
11  ?ldworkflow     ldwpo:firstLdwStep  ?step             .
12  ?step           (ldwpo:nextStep)*   ?linkedStep       .
13  ?linkedStep     ldwpo:inputDataset  ?inputDataset     .
14  ?inputDataset   ldwpo:name          ?nameDataset      .
15 } ORDER BY ?project
```

5.2.10 What are the steps done in this stage?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?ldWorkflow ?activity ?linkedStep WHERE {
8   ?project        rdf:type            ldwpo:LDWProject  .
9   ?activity       rdf:type            ldwpo:Activity    .
10  ?step           rdf:type            ldwpo:LDWStep     .
11  ?ldWorkflow     rdf:type            ldwpo:LDWorkflow  .
12  ?project        ldwpo:name          ?nameProject      .
13  ?project        ldwpo:ldWorkflow    ?ldWorkflow       .
14  ?ldWorkflow     ldwpo:firstLdwStep  ?step             .
15  ?step           (ldwpo:nextStep)*   ?linkedStep       .
16  ?linkedStep     ldwpo:task          ?task             .
17  ?activity       ldwpo:task          ?task             .
18 } ORDER BY ?project ?activity
```

5.2.11 What is the previous step of this step?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?ldWorkflow ?linkedStep ?previousStep WHERE {
8   ?project       rdf:type            ldwpo:LDWProject  .
9   ?step          rdf:type            ldwpo:LDWStep     .
10  ?linkedStep    rdf:type            ldwpo:LDWStep     .
11  ?previousStep  rdf:type            ldwpo:LDWStep     .
12  ?ldWorkflow    rdf:type            ldwpo:LDWorkflow  .
13  ?project       ldwpo:name          ?nameProject      .
14  ?project       ldwpo:ldWorkflow    ?ldWorkflow       .
15  ?ldWorkflow    ldwpo:firstLdwStep  ?step             .
16  ?ldWorkflow    ldwpo:ldwStep       ?linkedStep       .
17  ?ldWorkflow    ldwpo:ldwStep       ?previousStep     .
18  ?step          (ldwpo:nextStep)*   ?linkedStep       .
19  ?linkedStep    ldwpo:previousStep  ?previousStep     .
20 } ORDER BY ?project
```

5.2.12 What is the next step of this step?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?ldWorkflow ?linkedStep ?nextStep WHERE {
8   ?project       rdf:type            ldwpo:LDWProject  .
9   ?step          rdf:type            ldwpo:LDWStep     .
10  ?linkedStep    rdf:type            ldwpo:LDWStep     .
11  ?previousStep  rdf:type            ldwpo:LDWStep     .
12  ?ldWorkflow    rdf:type            ldwpo:LDWorkflow  .
13  ?project       ldwpo:name          ?nameProject      .
14  ?project       ldwpo:ldWorkflow    ?ldWorkflow       .
15  ?ldWorkflow    ldwpo:firstLdwStep  ?step             .
16  ?ldWorkflow    ldwpo:ldwStep       ?linkedStep       .
17  ?ldWorkflow    ldwpo:ldwStep       ?nextStep         .
18  ?step          (ldwpo:nextStep)*   ?linkedStep       .
19  ?linkedStep    ldwpo:nextStep      ?nextStep         .
20 } ORDER BY ?project
```

5.2.13 What is the format of the dataset?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameDataset ?format WHERE {
8    {
9      SELECT DISTINCT ?dataset ?format WHERE {
10       ?dataset    rdf:type            ldwpo:Dataset   .
11       ?step       rdf:type            ldwpo:LDWStep   .
12       ?format     rdf:type            ldwpo:Format    .
13       ?step       ldwpo:inputDataset  ?dataset        .
14       ?dataset    ldwpo:format        ?format         .
15     }
16   }
17   UNION
18   {
19     SELECT DISTINCT ?dataset ?format WHERE {
20       ?dataset    rdf:type            ldwpo:Dataset   .
21       ?step       rdf:type            ldwpo:LDWStep   .
22       ?format     rdf:type            ldwpo:Format    .
23       ?step       ldwpo:outputDataset ?dataset        .
24       ?dataset    ldwpo:format        ?format         .
25     }
26   }
27 } ORDER BY ?dataset
```

5.2.14 What are the project's goals?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameProject ?goal WHERE {
8    ?project  rdf:type     ldwpo:LDWProject .
9    ?project  ldwpo:name   ?nameProject     .
10   ?project  ldwpo:goal   ?goal            .
11 }
```

5.2.15 What is the license of the dataset?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?nameDataset ?nameLicense WHERE {
8    ?dataset  rdf:type       ldwpo:Dataset .
9    ?license  rdf:type       ldwpo:License .
10   ?dataset  ldwpo:name     ?nameDataset  .
11   ?dataset  ldwpo:license  ?license      .
12   ?license  ldwpo:name     ?nameLicense  .
13 }
```

5.2.16 What is the kind of task of this step?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?step ?task ?taskDescription WHERE {
8    ?step  rdf:type          ldwpo:LDWStep      .
9    ?task  rdf:type          ldwpo:Task         .
10   ?step  ldwpo:task        ?task              .
11   ?task  ldwpo:description  ?taskDescription  .
12 }
```

5.2.17 what are the tasks that this tool is applied?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?task ?tool WHERE {
8    ?step  rdf:type     ldwpo:LDWStep  .
9    ?task  rdf:type     ldwpo:Task     .
10   ?tool  rdf:type     ldwpo:Tool     .
11   ?step  ldwpo:task   ?task          .
12   ?step  ldwpo:tool   ?tool          .
13 }
```

5.2.18 Where is the input dataset stored?

```
1  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?dataset ?location WHERE {
8    ?dataset   rdf:type           ldwpo:Dataset   .
9    ?step      rdf:type           ldwpo:LDWStep   .
10   ?location  rdf:type           ldwpo:Location  .
11   ?step      ldwpo:inputDataset ?dataset        .
12   ?dataset   ldwpo:location     ?location       .
13 }
```

5.2.19 Where is the output dataset stored?

```
1  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:  <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?dataset ?location WHERE {
8    ?dataset   rdf:type              ldwpo:Dataset  .
9    ?step      rdf:type              ldwpo:LDWStep  .
10   ?location  rdf:type              ldwpo:Location .
11   ?step      ldwpo:outputDataset   ?dataset       .
12   ?dataset   ldwpo:location        ?location      .
13 }
```

5.2.20 When was the step executed?

```
1  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:  <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?stepExecution ?startedDate WHERE {
8    ?stepExecution  rdf:type           ldwpo:LDWStepExecution  .
9    ?stepExecution  ldwpo:startedDate  ?startedDate            .
10 }
```

5.2.21 When did the workflow start?

```
1  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:  <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?workflowExecution ?startedDate WHERE {
8    ?workflowExecution  rdf:type           ldwpo:LDWorkflowExecution  .
9    ?workflowExecution  ldwpo:startedDate  ?startedDate               .
10 }
```

5.2.22 When did the workflow finish?

```
1  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:  <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?workflowExecution ?endedDate WHERE {
8    ?workflowExecution  rdf:type         ldwpo:LDWorkflowExecution  .
9    ?workflowExecution  ldwpo:endedDate  ?endedDate                 .
10 }
```

5.2.23 Who are the people that contribute in this project?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?nameProject ?nameContributor WHERE {
8   ?project            rdf:type                 ldwpo:LDWProject    .
9   ?project            ldwpo:name               ?nameProject        .
10  ?project            ldwpo:ldWorkflow         ?ldworkflow         .
11  ?ldworkflow         ldwpo:firstLdwStep       ?step               .
12  ?ldworkflow         ldwpo:ldWorkflowExecution ?workflowExecution .
13  ?step               (ldwpo:nextStep)*        ?linkedStep         .
14  ?linkedStep         ldwpo:ldwStepExecution   ?stepExecution      .
15  ?workflowExecution  ldwpo:ldwStepExecution   ?stepExecution      .
16  ?stepExecution      ldwpo:contributor        ?contributor        .
17  ?contributor        ldwpo:name               ?nameContributor    .
18 } ORDER BY ?project
```

5.2.24 Who is responsible for this step?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?stepExecution ?nameContributor WHERE {
8   ?stepExecution  rdf:type           ldwpo:LDWStepExecution  .
9   ?stepExecution  ldwpo:contributor  ?contributor            .
10  ?contributor    ldwpo:name         ?nameContributor        .
11 }
```

5.2.25 How is the input dataset stored?

```
1 PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl:   <http://www.w3.org/2002/07/owl#>
3 PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
4 PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7 SELECT DISTINCT ?dataset ?format WHERE {
8   ?dataset    rdf:type            ldwpo:Dataset   .
9   ?step       rdf:type            ldwpo:LDWStep   .
10  ?format     rdf:type            ldwpo:Format    .
11  ?step       ldwpo:inputDataset  ?dataset        .
12  ?dataset    ldwpo:format        ?format         .
13 }
```

5.2.26 How is the output dataset stored?

```
1  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl:  <http://www.w3.org/2002/07/owl#>
3  PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX ldwpo: <http://ldwpo.aksw.org/terms/1.0/>
6
7  SELECT DISTINCT ?dataset ?format WHERE {
8    ?dataset      rdf:type            ldwpo:Dataset   .
9    ?step         rdf:type            ldwpo:LDWStep   .
10   ?format       rdf:type            ldwpo:Format    .
11   ?step         ldwpo:outputDataset ?dataset        .
12   ?dataset      ldwpo:format        ?format         .
13 }
```

5.3 Use cases consideration

In this section, we present qualisConversion and qualisInterlinking LDWProjects as real-world use cases of LDWPO. They are represented by Figure 4 and Figure 5, which show the partial reports generated by the reportgen tool (see subsection 6.1). These LDWProjects were executed in 25-04-2015, using the ldStackWorkflow tool (see subsection 6.2). As result of the executions, Qualis data is available as Linked Open Data (see the VoID description listed in subsection 6.3).

5.3.1 qualisConversion LDWProject

1) PROJECT IDENTIFICATION	
1.1) Name	QualisBrasil Project
1.2) Creator	Sandro Rautenberg
1.3) Homepage	http://lodunicentro.aksw.org/
1.5) Purpose(s)	Offering Qualis Periodical Index according to the principles Linked Open Data
1.6) Description	QualisBrasil is part of http://lod.unicentro.br endpoint. It aims to aid researchers to collect Linked Open Data about Qualis Index in bibliometrics or scientometrics studies. Available data: Periodical ISSN, Periodical Name, Knowledge Field, Qualis Index, and Year.

2) PLANNING WORKFLOW	
Name	Planning Workflow of Maintain QualisBrasil
What should be done	Workflow applied to create linked dataset of Qualis Periodicals Index (all years), in a automatized way.
Precondition	1. Availability of the data (sql dump) 2. Running system with installed Ubuntu and all necessary tools (i.e. Sparqlify, Virtuoso, and all the scripts)
Postcondition	QualisBrasil Graph is created and maintained.
Step(s)	
step (1)	ldwStep_extracting_qualisbrasil_from_legacy_system
task classification	[extract data from legacy systems/databases] [select data sources] [ensure that the temporal dimension is preserved within the data model] [ensure unique identifiers for each resource] [automatize the extraction process in a way that enables automatic updates]
output dataset(s)	name : evaluations.csv
tool	name: MySQL location: /usr/share/mysql
what should be done	Extracting QualisBrasil from Legacy System, using a configuration file. Converting data to CSV.
command	real/QualisBrasilProject/bin/extractingFromLegacySystem.sh
step (2)	ldwStep_extracting_qualisbrasil_applying_sparqlify
task classification	[create a vocabulary that fits your data] [reuse existing RDF vocabularies]
input dataset(s)	name : evaluations.csv
output dataset(s)	name : evolutions.nt
tool	name: Sparqlify location: /usr/share/lib/sparqlify/
what should be done	Using Sparqlify and a configuration file tp convert data to N-Triples.
command	real/QualisBrasilProject/bin/applyingSparqlify.sh
tool configuration	name : mappingQualis.sml
step (3)	ldwStep_storing_qualisbrasil
task classification	[ensure the high availability of the data for real-time applications by using state-of-the-art triplestores] [automatize the process in a way that enables automatic storing]
input dataset(s)	name : QualisBrasil.nt
output dataset(s)	name : Qualis Brasil Graph
what should be done	In this step the file QualisBrasil.nt is uploaded to Virtuoso by executing the command based on the toolConfiguration file.
command	real/commons/savingIntoVirtuoso.sh real/Graphs/QualisBrasil/QualisBrasil.nt http://lod.unicentro.br/QualisBrasil/
step (4)	ldwStep_finalMessage_qualisbrasil
what should be done	it shows a message informing the workflow is finished.
command	echo "Finito Sparqlify..."

Figure 4: Partial report of qualisConversion LDWProject

After applying the qualisConversion LDWProject, we got 698668 entities of Qualis uploaded into the Linked Data cloud, according the sparql query and results showing below.

```
1  #Number of entities per year in the Graph
2  PREFIX dc:   <http://purl.org/dc/elements/1.1/>
3  PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5  PREFIX owl: <http://www.w3.org/2002/07/owl#>
6
7  SELECT ?yearEvaluation count(?journal) as ?NUM_EVALUATIONS_PER_YEAR WHERE {
8    ?evaluation rdf:type rdf:Class .
9    ?evaluation qualis:hasJournal ?journal .
10   ?evaluation qualis:hasYear ?yearEvaluation .
11   ?yearEvaluation rdf:type rdf:Class .
12   FILTER (str(?yearEvaluation) != "http://lod.unicentro.br/QualisBrasil/YearEvaluation"
        )
13 } GROUP BY ?yearEvaluation ORDER BY ?yearEvaluation
```

yearEvaluation	NUM_EVALUATIONS_PER_YEAR
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2005	35020
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2006	35020
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2007	35020
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2008	54233
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2009	54233
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2010	54233
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2011	107429
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2012	107429
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2013	107429
http://lod.unicentro.br/QualisBrasil/YearEvaluation_2014	108622

5.3.2 qualisInterlinking LDWProject

1) PROJECT IDENTIFICATION	
1.1) Name	Interlinking DBPedia to Qualis Project
1.2) Creator	Sandro Rautenberg
1.3) Homepage	http://lodunicentro.aksw.org/
1.5) Purpose(s)	Interlinking Qualis Brasil to DBPedia.
1.6) Description	qualisInterlinking LDWProject converts the Qualis data from a 4 to a 5 Stars Open Data.

2) PLANNING WORKFLOW	
Name	**Planning Workflow of interlinking QualisBrasil with LIMES**
What should be done	This LDWorkflow comprises three LDWSteps: 1) uses LIMES to interlink Qualis with DBpedia data; 2) reuses the store LDWStep from qualisConversion LDWProject; and 3) provides user with feedback to the LDWorkflowExecution
Precondition	qualisConversion LDWProject must be executed.
Postcondition	Qualis data is interlinked and uploaded in a triple store.
Step(s)	
step (1)	ldwStep_interlinking_qualisbrasil_applying_LIMES
task classification	[define related open datasets to be interlinked] [create consumption patterns which specify the data selection and the method of interlinking] [automatize the interlinking process in a way that enables automatic updates]
input dataset(s)	**name** : Qualis Brasil Graph **name** : DBpedia
output dataset(s)	**name** : QualisBrasil.nt
what should be done	applying LIMES and a tool configuration.
command	real/QualisBrasil_to_DBpedia_InterlinkingProject/bin/interlinkingQualisToDBPedia.sh
tool configuration	**name** : linkingQualisToDBpedia.xml
step (2)	ldwStep_storing_qualisbrasil
task classification	[ensure the high availability of the data for real-time applications by using state-of-the-art triplestores] [automatize the process in a way that enables automatic storing]
input dataset(s)	**name** : QualisBrasil.nt
output dataset(s)	**name** : Qualis Brasil Graph
what should be done	In this step the file QualisBrasil.nt is uploaded to Virtuoso by executing the command based on the toolConfiguration file.
command	real/commons/savingIntoVirtuoso.sh real/Graphs/QualisBrasil/QualisBrasil.nt http://lod.unicentro.br/QualisBrasil/
step (3)	ldwStep_finalMessage_interlinking_qualisbrasil
what should be done	only shows a message
command	echo "Finito LIMES..."

Figure 5: Partial report of qualisInterlinking LDWProject

After applying the qualisInterlinked LDWProject, we got 4036 entities of Qualis interlinked to the some DBpedia resources.

```
1  #Number of entities interlinked to DBpedia in 01-04-2015.
2  PREFIX qualis:   <http://lod.unicentro.br/QualisBrasil/>
3  PREFIX dc:    <http://purl.org/dc/elements/1.1/>
4  PREFIX rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
6  PREFIX owl: <http://www.w3.org/2002/07/owl#>
7
8  SELECT count(?journal) AS ?NUM_RESOURCES_INTERLINKED_TO_DBPEDIA WHERE {
9    ?dbpedia owl:sameAs  ?journal .
10   {
11     SELECT DISTINCT ?journal WHERE {
12       ?s qualis:hasJournal ?journal .
13     }
14   }
15 }
```

NUM_RESOURCES_INTERLINKED_TO_DBPEDIA
4036

With the use cases, a `5-star` dataset was generated by our automated process using LDWPO. This dataset is published on datahub[9] and public available at `http://lodkem.led.ufsc.br:8890/sparql`[10]. It comprises more than *600 thousand* interlinked entities that can be queried and used to facilitate the bibliometric and scientometric research in Brazil. Moreover, as all project information and implemented systems are published at `https://github.com/AKSW/ldwpo`.

---

[9]`http://datahub.io/dataset/qualisbrasil`
[10]graph name in triple store: "http://lod.unicentro.br/QualisBrasil"

6 Other results

Here, we list the another results by developing LDWPO. We developed computational tools to support the execution of real-world use cases on applying the ontology. In addition, a real-world RDF dataset is available as Linked Open Data representing Qualis data, a dataset used by the Brazilian Scientific Community for promoting scientometric studies (see the VoID description of this dataset).

6.1 reportgen Tool

We developed a report tool for generating an LDWProject document. This tool is named reportgen
Tool and is available at `https://github.com/AKSW/ldwpo/tree/master/tools`).

To use reportgen go to tool/reportgen/ folder and run "$ java -jar reportgen.jar".

The command will output a help message.

6.2 ldStackWorkflow Tool

We developed a workflow tool for executing an LDWProject in an automatic way. This tool is named ldStackWorkflow Tool and is available at `https://github.com/AKSW/ldwpo/tree/master/tools`).

To use ldStackWorkflow go to tool/ldStackWorkflow/ folder and run "$ java -jar ldStackWorkflow.jar".

The command will output a help message.

6.3 Qualis Brasil Linked Open Data dataset

```
1  @prefix dcat: <http://www.w3.org/ns/dcat#> .
2  @prefix dcterms: <http://purl.org/dc/terms/> .
3  @prefix dctypes: <http://purl.org/dc/dcmitype/> .
4  @prefix foaf: <http://xmlns.com/foaf/0.1/> .
5  @prefix freq: <http://purl.org/cld/freq/> .
6  @prefix owl: <http://www.w3.org/2002/07/owl#> .
7  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
8  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
9  @prefix sd: <http://www.w3.org/ns/sparql-service-description#> .
10 @prefix sio: <http://semanticscience.org/resource/> .
11 @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
12 @prefix uniprot: <http://purl.uniprot.org/uniprot/> .
13 @prefix void: <http://rdfs.org/ns/void#> .
14 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
15
16 <http://lod.unicentro.br/QualisBrasil/>
17   dcterms:title "Qualis Brasil" ;
18   dcterms:description "Qualis Brasil encompasses indirect scores for research papers in
          journals, according to 48 knowledge fields, such as computer science, chemistry
          , medicine, among others. It is used to rank post-graduate programs, research
          proposals, or individual research scholarships. A typical entry of Qualis
          consists of ISSN, journal name, knowledge field, and qualified journal score.
          Moreover, the journal can be related to one or more scores, based on the
          relevance of the journal to the knowledge field." ;
19   dcterms:publisher  <Sandro_Rautenberg>;
20   dcterms:issued "2015-04-01T00:00:00.000+01:00"^^xsd:dateTime ;
21   dcterms:license <http://creativecommons.org/publicdomain/zero/1.0/> ;
22   dcterms:subject <http://dbpedia.org/page/Scientometrics> ;
23   dcterms:subject <http://dbpedia.org/page/Bibliometrics>;
24   dcterms:source <http://qualis.capes.gov.br/webqualis/principal.seam> ;
25   dcterms:accrualPeriodicity freq:annual ;
26
27   void:sparqlEndpoint <http://lodkem.led.ufsc.br:8890/sparql> ;
28   void:exampleResource <http://lod.unicentro.br/QualisBrasil/Journal_0004-5411
          _KnowledgeField_2_YearEvaluation_2014_Qualis_A1> ;
29   void:exampleResource <http://lod.unicentro.br/QualisBrasil/Journal_0004-5411> ;
30   void:exampleResource <http://lod.unicentro.br/QualisBrasil/KnowledgeField_2> ;
31   void:exampleResource <http://lod.unicentro.br/QualisBrasil/Qualis_A1> ;
32   void:exampleResource <http://lod.unicentro.br/QualisBrasil/YearEvaluation_2014> ;
33   void:triples 3590448 ;
34   void:entities 698669 ;
35   void:properties 9 ;
36 a void:Dataset ;.
37
38 <Sandro_Rautenberg>
39   rdfs:label "Sandro Rautenberg" ;
40   foaf:mbox <mailto:srautenberg@unicentro.br> ;
41   foaf:homepage <http://buscatextual.cnpq.br/buscatextual/visualizacv.do?metodo=
          apresentar&id=K4700255Z6>;  a foaf:Person; .
```

7 License for using LDWPO

This ontology will be published under the license "Attribution-ShareAlike CC BY-SA".

Acknowledgment

References

[1] Sören Auer, Lorenz Bühmann, Christian Dirschl, Orri Erling, Michael Hausenblas, Robert Isele, Jens Lehmann, Michael Martin, Pablo N. Mendes, Bert van Nuffelen, Claus Stadler, Sebastian Tramp, and Hugh Williams. Managing the life-cycle of linked data with the LOD2 stack. In *Proceedings of International Semantic Web Conference (ISWC 2012)*, 2012.

[2] Sören Auer. Introduction to lod2. In Sören Auer, Volha Bryl, and Sebastian Tramp, editors, *Linked Open Data – Creating Knowledge Out of Interlinked Data*. Springer-Verlag, 2014.

[3] Asunción Gomez-Perez, Mariano Fernandez-Lopez, and Oscar Corcho, editors. *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web (2nd edition)*. Springer-Verlag, Heidelberg, 2007.

[4] Natalya F. Noy and Deborah L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical Report SMI-2001-0880, Stanford University School of Medicine, 2001. This guide describes a common methodology for ontology-development based on declerative frame-based systems. Upshot: there is no single correct ontology for any domain.

[5] J.F. Sowa and John A. Zachman. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, 31(3):590–616, 1992.

[6] York Sure and Rudi Studer. On-To-Knowledge methodology. In John Davies, Dieter Fensel, and Frank van Harmelen, editors, *On-To-Knowledge: Semantic Web enabled Knowledge Management*, chapter 3, pages 33–46. J. Wiley and Sons, 2002.