Semantic Web 0 (0) 1–12
IOS Press

The Publishing Workflow Ontology (PWO)

Editor(s): Victor de Boer, Vrije Universiteit Amsterdam, Netherlands; Agnieszka Ławrynowicz, Poznań University of Technology, Poland
Solicited review(s): Boris Villazón-Terrazas, iSOCO, Spain; Ondřej Zamazal, Prague University of Economics, Poland; Raúl Palma, Poznań Supercomputing and Networking Center, Poland

Aldo Gangemi [a], Silvio Peroni [b,*], David Shotton [c], Fabio Vitali [d]
[a] *STLab-ISTC, Consiglio Nazionale delle Ricerche (Italy)*
Laboratoire d'Informatique de Paris Nord, Université Paris 13 (France)
aldo.gangemi@cnr.it
[b] *Department of Computer Science and Engineering, University of Bologna (Italy)*
silvio.peroni@unibo.it
[c] *Oxford e-Research Centre, University of Oxford (UK)*
david.shotton@oerc.ox.ac.uk
[d] *Department of Computer Science and Engineering, University of Bologna (Italy)*
fabio.vitali@unibo.it

Abstract. In this paper we introduce the *Publishing Workflow Ontology* (*PWO*), i.e., an OWL 2 DL ontology for the description of workflows that is particularly suitable for formalising typical publishing processes such as the publication of articles in journals. We support the presentation with a discussion of all the ontology design patterns that have been reused for modelling the main characteristics of publishing workflows. In addition, we present two possible application of PWO in the publishing and legislative domains.

Keywords: PWO, ODP, publishing process, workflow description

1. Introduction

Keeping track of publication processes is a crucial task for publishers. This action allows them to produce statistics on their goods (e.g., books, authors, editors) and to understand whether and how their production changes over time. Organisers of particular events, such as academic conferences, have similar needs. Tracking the number of submissions in the current edition of a conference, the number of accepted papers, the review process, etc., are important statistics that can be used to improve the review process in future editions of the conference.

Some communities have started to publish data, e.g., the Semantic Web Dog Food[1] and the Semantic Web Journal[2], as RDF statements in the Linked Open Data Cloud, that allow software agents and applications to check and reason on them, and to infer new information. However, the description of processes, for instance the peer-review process or the publishing process, is something that is not currently handled – although sources of related raw data exist (e.g., EasyChair metadata). Furthermore, a flexible model for describing these data, developed according to guidelines for guaranteeing its reusability in different domains, is also needed. Having these types of data publicly available and described according to such model would result in:

– increasing transparency of the (publishing) process;

*Corresponding author. E-mail: silvio.peroni@unibo.it.
[1] Semantic Web Dog Food: http://data.semanticweb.org.
[2] Semantic Web Journal: http://semantic-web-journal.com.

– enabling the use of such data for statistical analysis;
– facilitating the integration, alignment and adaptation of both the data and the model according to the needs and constraints of different domains (publishing, academic conferences, research funding, etc.).

In this paper, which extends our work presented at the *5th International Workshop on Ontology and Semantic Web Patterns* [10][3], we introduce the *Publishing Workflow Ontology* (*PWO*), that we developed in order to reach the aforementioned goals. This ontology is one of the *Semantic Publishing and Referencing* (*SPAR*) *Ontologies*[4] [20] (which have been created for the description of different aspects of the publishing domain), and allows one to describe the logical steps in a workflow, as for example the process of publication of a document. Each step may involve one or more events that take place at a particular phase of the workflow (e.g., authors are writing the article, the article is under review, a reviewer suggests to revise the article, the article is in printing, the article has been published, etc.). This ontology has been developed in order to allow its use with other SPAR Ontologies as well as other models and existing data.

The rest of the paper is organised as follows. In Section 2 we discuss some related works on workflows within the Semantic Web domain. In Section 3 we provide the definitions of workflow we have used as starting point for modelling our ontology, and discuss the use of some existing ontology design patterns for addressing the modelling issues related to the main characteristics of workflows. In Section 4 we introduce PWO, describing how it extends the aforementioned patterns in order to handle the main components of workflows. In Section 5 we show how to use PWO for modelling data related to the publication process of an article of the Semantic Web Journal and to the codification process of laws in US legislation. Finally, in Section 6 we conclude the paper sketching out some future works.

2. Workflows and the Semantic Web

The use and managing of workflows is a relevant aspect in different applicative domains. For instance, formalised approaches for workflow definitions and executions have been adopted in industrial applications for managing and delivering products[5], XML-based scholarly publishing [19], and Wiki-oriented publication mechanisms[6].

In the last years the Semantic Web community have also started on working and proposing models for the formalisation and description of generic workflows, and have shown several applications of these models/theories within the publishing domain. Maybe the first huge-impact project on these topic has been Wf4Ever (STREP FP7-ICT-2007-6 270192)[7] [12]. This project addresses challenges related to the preservation of scientific experiments through the definition of models and ontologies for describing scientific experiments, to the collection of best practices for the creation and management of *Research Objects*[8] [2], and to the analysis and management of decay in scientific workflows.

As already stated, one of the outcomes of the project has been the proposal for workflow-centric Research Objects [1], i.e., an OWL ontology[9] for linking together scientific workflows, the provenance of their executions, interconnections between workflows and related resources (e.g., datasets, publications, etc.), and social aspects related to such scientific experiments.

Another interesting proposal for describing workflows is the work done by Garijo and Gil [11]. In this work, they describe a framework to publish *computa-*

[3] This paper was selected by the members of the organising committee as one of the best papers of the workshop for being part of a fast-track submission to the Semantic Web journal's Special Issue on Ontology Design Patterns. However, more than proposing a new pattern, our work extends the published workshop paper and focuses on the reuse of design patterns for developing an ontology for publishing workflows. The additional material provided in this revised version concerns: the description of the design pattern adopted, which has been extended in order to clarify several parts and that now includes a precise specification of the requirements for such model; the extension of the section describing our ontology, that now includes some statistics and definitions that were not present in the original article, as well as the introduction of several new parts of the ontology that have been added as consequence of reviews and comments we received during the workshop; the introduction of a new example of use in the legislative domain, and an example of SPARQL queries for answering some relevant questions within the publishing domain.

[4] SPAR Ontologies website: http://www.sparontologies.net.

[5] SDL LiveContent 2014: http://docs.sdl.com/LiveContent/content/en-US/SDL%20LiveContent%20full%20documentation-v1/GUID-867B6863-ADAD-40C9-A3F7-775FE29FAFF3

[6] http://extensions.xwiki.org/xwiki/bin/view/Extension/XWiki+Publication+Workflow+Application

[7] Wf4Ever project homepage: http://www.wf4ever-project.org.

[8] Research Object website: http://www.researchobject.org.

[9] Research Object OWL ontology: http://purl.org/wf4ever/ro.

tional workflows, which includes the specification of a particular OWL ontology, i.e., the *Open Provenance Model for Workflows* (*OPMW*)[10], for the description of workflow traces and their templates. Along the lines of the aforementioned work, the same authors recently published the *Ontology for Provenance and Plans* (*P-Plan*)[11]. P-Plan is an OWL 2 DL ontology that extends the *Provenance Ontology* [16] in order to represent the plans that guided the execution of scientific processes, describing how such plans are composed and their correspondence to provenance records that describe the execution itself.

Finally, among the other proposals for describing workflows, it is worth mentioning the OWL ontology proposed by Sebastian *et al.* [28] for describing generic workflows, which reuses existing ontologies such as the *Change and Annotations Ontology* (*ChAO*) [18], and the *SCUFL2 Core ontology*[12] that has been used to describe workflows in *Taverna*[13], an open source and domain-independent Workflow Management System [29].

The first stable version of the *Publishing Workflow Ontology* (*PWO*) has been released in 2010 as part of the SPAR Ontologies. It was pattern-based but it did not include any of the entities introduced in the aforementioned ontologies, since some of them were (as far as we knew at the time) either not available or not released in stable form. Some of the modelling ideas presented in papers have been considered and in many cases followed.

Of course, since the first release, PWO has been extended and the version presented in this article is the result of such evolution. However, in order to foster its reuse and interoperability with the aforementioned OWL ontologies, an initial alignment path with PWO has been created and it is available online[14].

3. Foundational material: design patterns

In order to design an ontology for modelling (publishing) workflows, we have to understand what are the minimal characteristics that such ontology should address, and if we can reuse existing modelling solutions. In order to gather minimal requirements, we take into account both general definitions of workflows, and existing workflow models, as, e.g., generalised in [27]. Minimal requirements will then be used to retrieve or create appropriate ontology design patterns [9], which have proved to substantially improve ontology design quality [3], due to their ability to address the semantics of requirements, and to their modularity.

Definitions are quite convergent, e.g., the Oxford Dictionaries site defines workflow as follows:

> "The sequence of industrial, administrative, or other processes through which a piece of work passes from initiation to completion."[15]

From this definition it is possible to identify some important characteristics (listed as *R1*, ..., *Rn* requirements) of any workflow, i.e., the fact that:

- it *involves* a *sequence* of processes **[R1]**;
- each process allows one to initiate and then complete a piece of work during a specifiable *time interval* **[R2]**.

The definition of the SearchCIO website is still more specific:

> "Workflow is a term used to describe the tasks, procedural steps, organisations or people involved, required input and output information, and tools needed for each step in a business process."[16]

From this definition we can spot other crucial aspects:

- the structure of a workflow is organised in procedural steps **[R3]**;
- each step *describes* tasks **[R4]**;
- each task is performed by organisations or people **[R5]**;
- each task *requires* some input information in order to *produce* an output. **[R6]**.

Control flow patterns [27] address additional requirements on the possible structure of the steps:

- sequencing **[R7]**;
- parallel splitting **[R8]**;
- synchronization **[R9]**;
- exclusive choice **[R10]**;
- simple merging **[R11]**.

[10] Open Provenance Model for Workflows: http://www.opmw.org/ontology/.
[11] Ontology for Provenance and Plans: http://purl.org/net/p-plan#.
[12] http://ns.taverna.org.uk/2010/scufl2
[13] http://www.taverna.org.uk
[14] http://purl.org/spar/pwo-alignment
[15] http://www.oxforddictionaries.com/definition/english/workflow
[16] http://searchcio.techtarget.com/definition/workflow

Using the eleven requirements extracted from definitions and models, we can retrieve ontology design patterns that most closely address (typically in an abstract way) the notions related to workflows.

3.1. Participation

The *participation pattern*[17] is a simple pattern to describe processes, events, or states (in the class *Event*), and to specify the various objects (in the class *Object*) that participate in these events.

This pattern seems to be very useful to define events or actions within workflow executions, involving people, organisations, places, and other objects as participants [R5], as well as to link actions to the expected steps in a workflow [R4].

3.2. Sequence

The *sequence pattern*[18] is an abstract pattern that generalizes over arbitrary (spatial, temporal, conceptual, logical) sequential orderings. It can be used between tasks, processes or time intervals, in order to define sequences of such objects through direct (i.e., *directlyFollows* and *directlyPrecedes*) or transitive relations (i.e., *follows* and *precedes*), which are linked by means of the *transitive reduction* logical pattern (each direct relation is a subrelation of a transitive relation, e.g., `directlyFollow owl:subPropertyOf follows`).

For workflows, it results useful to describe the logical organisation of steps in a workflow [R1][R7], or the temporal ordering of actions or events in a workflow execution.

3.3. Control flow and plan execution

The *control flow* pattern[19] is an OWL representation of some of the control flow patterns defined in the Workflow Patterns[20] repository (cf. [27]). The control flow pattern imports other patterns: *task execution, task role, sequence, participation*[21]. The pattern represents either action tasks (typically event types), or control tasks (e.g., branching, concurrency, looping, cf. [R8-R11]). Their sequential ordering [R7] is represented by means of the *sequence pattern*. *Tasks* are distinct from *actions*, which are supposed to be executed based on the task structure:

```
Individual: submission
    Types: Action

Individual: review
    Types: Action

Individual: submission-task
    Types: ActionTask
    Facts: taskex:isExecutedIn submission

Individual: reviewing-task
    Types: ActionTask
    Facts:
        taskex:isExecutedIn review ,
        sequence:follows submission-task
```

The control flow pattern is supposed to be composed with the *basic plan*[22] pattern, composed of the *basic plan description*[23] and the *basic plan execution*[24] patterns. Basic plan reuses the foundational *descriptions and situations pattern* to relate task compositions (*plans*) and organised actions (*plan executions*). A comprehensive presentation is provided in [8] (the general axiomatization of plan models and execution, and control flows), in [25] (a description of the above mentioned patterns), and in [7] (an application of plan models to social ontology and norms). Since OWL semantics is open world, in order to enforce sequencing constraints on actions classified by a task, rules should be defined which provide integrity checks, cf. the *error ontology* below.

These patterns are needed to describe the kinds of steps (the term used here for *tasks*) in publishing workflows [R3, R4]. The action and control tasks from the control flow pattern are not specialised in the publishing workflow pattern, because they are expected to work as they are (by typing the steps according to their workflow semantics), when the need for control flows emerges in a planned workflow.

3.4. Time-indexed situation

The *time-indexed situation pattern*[25] allows the description of a situation – i.e., the class *TimeIndexedSit-*

[17] http://www.ontologydesignpatterns.org/cp/owl/participation.owl

[18] http://www.ontologydesignpatterns.org/cp/owl/sequence.owl

[19] http://www.ontologydesignpatterns.org/cp/owl/controlflow.owl

[20] The Workflow Patterns page is: http://www.workflowpatterns.com.

[21] Similarly available from the ontologydesignpatterns.org portal.

[22] http://www.ontologydesignpatterns.org/cp/owl/basicplan.owl

[23] http://www.ontologydesignpatterns.org/cp/owl/basicplandescription.owl

[24] http://www.ontologydesignpatterns.org/cp/owl/basicplanexecution.owl

[25] http://www.ontologydesignpatterns.org/cp/owl/timeindexedsituation.owl

uation presented as a view on a set of entities linked to it through the property *isSettingFor* – that is explicitly indexed at some time specifiable through the property *atTime* linking a time interval (i.e., an instance of the class *TimeInterval*).

This pattern combines perfectly with plan execution in order to provide a temporal ordering to actions [R2] organised into a plan.

3.5. Error Ontology

The *Error Ontology*[26] is a unit test that produces an inconsistent model if a particular (and incorrect) situation happens. It works by means of a data property, *error:hasError*, that denies its usage for any resource, as shown as below (in Manchester Syntax [13]):

```
DataProperty: error:hasError
  Domain: error:hasError exactly 0
  Range: xsd:string
```

A resource that has an error makes the ontology inconsistent, since its domain is "all those resources that do not have any *error:hasError* assertion".

This model is very useful in our context in order to define constraints on the input/output objects needed by the steps of a workflow [R6]. For instance, we could use it to deny the specification of a certain object as input of a step if it will be produced only as output of one of the following steps.

4. PWO: the Publishing Workflow Ontology

In order to accommodate workflow requirements, we developed the *Publishing Workflow Ontology*[27] (*PWO*)[28], which is entirely based on the ontology patterns introduced in Section 3. This ontology enables the description of the logical steps in a workflow, as, e.g., the process of publication of a document. Each step may involve one or more events (or actions) that take place in a particular phase of the workflow (e.g., authors are writing the article, the article is under review, a reviewer suggests to revise the article, the article is in printing, the article has been published, etc.). Currently (March 14, 2016) the ontology:

- imports eight ontologies (seven of which describing existing patterns);
- defines four new classes (*pwo:Workflow*, *pwo:WorkflowExecution*, *pwo:Step*, *pwo:Action*);
- defines twelve new object properties (*pwo:executes*, *pwo:hasStep*, *pwo:hasFirstStep*, *pwo:involvesAction*, *pwo:needs*, *pwo:produces*, and the related inverses);
- does not introduce new data properties.

As shown in Fig. 1, PWO is based on two main classes *pwo:Workflow* and *pwo:Step*. *pwo:Workflow* represents a sequence of connected tasks (i.e., steps, by using the property *pwo:hasFirstStep* and *pwo:hasStep*, which are ordered by means of *pwo:hasNextStep*). *pwo:Workflow* is a subclass of *plandesc:Plan*. We can use the class *pwo:WorkflowExecution* to represent the aggregate of the actions (class *pwo:Action*) executed in the steps defined in a workflow. We can then abstract from the execution of individual steps, by saying that a certain workflow execution *executes* a workflow[29]. The formal representation of the class *pwo:Workflow* and the related entities is introduced as follows (in Manchester Syntax):

```
Class: pwo:Workflow
    SubClassOf:
        plandesc:Plan,
        pwo:hasFirstStep some pwo:Step

ObjectProperty: pwo:hasStep
    SubPropertyOf: plandesc:definesTask
    Domain: pwo:Workflow
    Range: pwo:Step

ObjectProperty: pwo:hasFirstStep
    SubPropertyOf: pwo:hasStep
    Range:
        not (pwo:hasPreviousStep some pwo:Step)

ObjectProperty: pwo:hasNextStep
    SubPropertyOf: sequence:directlyPrecedes
    Domain: pwo:Step
    Range: pwo:Step
    InverseOf: pwo:hasPreviousStep

Class: WorkflowExecution
    SubClassOf:
        tisit:TimeIndexedSituation,
        planex:PlanExecution,
        pwo:executes some pwo:Workflow,
        pwo:involvesAction some
            (pwo:Action that taskex:executesTask
```

[26] http://www.essepuntato.it/2009/10/error

[27] http://purl.org/spar/pwo

[28] As already introduced in Section 2, in PWO we do not directly reuse any particular entity defined in the ontologies described in Section 2 on purpose. This was due to a particular design choice during the development of PWO, i.e., we wanted to rely only on a pattern-based supported ontology development, while the alignment to existing ontologies is available in a separate file from http://purl.org/spar/pwo-alignment.

[29] If we want to check if a workflow has been actually executed, we need to aggregate the rules that check the execution of all the individual steps from a workflow.

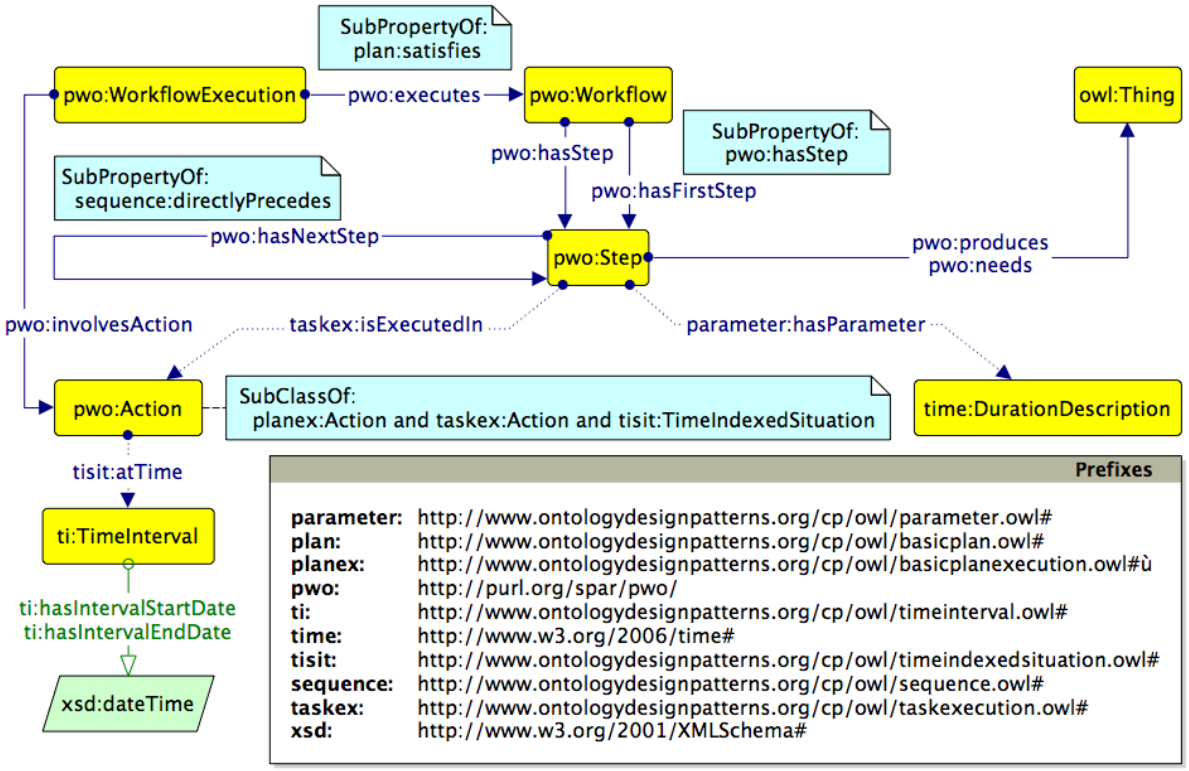

Fig. 1. Graffoo representation [6] of the Publishing Workflow Ontology (PWO).

```
                some pwo:Step)

ObjectProperty: pwo:executes
    SubPropertyOf:
        plan:satisfies
    Domain: pwo:WorkflowExecution
    Range: pwo:Workflow
```

The other main class of PWO is *pwo:Step*. A step is an atomic unit of a workflow, and subclass of *taskrole:Task*; it is characterised by (required) temporal parameters (*parameter:hasParameter*) describing the expected duration (*time:DurationDescription*) of executed actions, and it is associated with one or more (time-indexed) executed actions by using the property *taskex:isExecutedIn*. A workflow step usually involves some input information, material or energy needed to complete the step (i.e., *pwo:needs*), and some output information, material or energy produced by that step (i.e., *pwo:produces*). In the case of a publishing workflow, a step typically results in the creation of a publication entity, usually by the modification of another pre-existing publication entity, e.g., the creation of an edited paper from a rough draft, or of an HTML representation from an XML document. The formal representation of the class *pwo:Step* and its related properties is the following (in Manchester Syntax):

```
Class: pwo:Step
    SubClassOf:
        taskrole:Task,
        taskex:isExecutedIn only pwo:Action,
        plandesc:isTaskDefinedIn
            only pwo:Workflow,
        parameter:hasParameter
            exactly 1 time:DurationDescription

Class: pwo:Action
    SubClassOf:
        planex:Action,
        taskex:Action,
        tisit:TimeIndexedSituation

ObjectProperty: pwo:involvesAction
    Domain:
        pwo:WorkflowExecution
    Range: pwo:Action
    SubPropertyChain:
        pwo:executes o
        pwo:hasStep o
        taskex:isExecutedIn

ObjectProperty: pwo:needs
    Domain: pwo:Step
    Range: owl:Thing

ObjectProperty: pwo:produces
    Domain: pwo:Step
```

```
Range: owl:Thing
```

PWO has been implemented according to the aforementioned ontology patterns. As shown in Table 1[30], such patterns have been used as follows:

- *basic plan* to describe workflows as plans, and their executions;
- *time-indexed situation* to describe the actions executing a step, that involve a duration and that are characterised by events and objects (among which those that are needed for or produced by steps);
- *sequence* to define the order, in which steps appear within a workflow;
- *control flow* to describe the specialisation and nature of steps at planning time;
- *participation* to describe events (and eventually agents involved) taking part in the actions carried out according to the steps.

In addition, by means of the Error Ontology, we generate an inconsistency when the steps of a workflow are not arranged in the correct temporal order. In particular, an error is raised when a step requires (property *pwo:needs*) a particular object to be produced (e.g., from the property *pwo:produces*) as consequence of another sequent step. The following excerpt shows the implementation of this constraint in a SWRL rule [14]:

```
Step(?step1) , Step(?step2) ,
needs(?step1,?resource) ,
produces(?step2,?resource) ,
sequence:precedes(?step1,?step2)
  -> error:hasError(?step1,"A step cannot need a
       resource that will be produced by a
       following step"^^xsd:string)
```

It is worth mentioning that the final two properties introduced in Table 1, i.e., *pwo:needs* and *pwo:produces*, are associated to the description-level of the workflow, since they are related to the class *pwo:Step*. However, the particular specification of what is needed/produced by a step is not yet available when one develops a workflow description. Such specification will be provided dynamically at execution time and will depend strictly on the objects that will be really used during such execution. Basically speaking, by using PWO for modelling workflows, the statements that will be specified for a particular workflow execution will (and must) populate dynamically both the workflow execution and the related workflow description.

5. Applying PWO to the publishing and legislative domains

In the next subsections we show how to describe the process of publication of a journal article, as well as the process of codification laws of US legislation step by step. The examples apply PWO to existing data from the *Semantic Web Journal*[31] [15], DBPedia [17], and other SPAR ontologies, such as PSO [24], C4O [5], FaBiO and CiTO [23].

5.1. A typical publishing workflow of a journal article

From the publisher's perspective, the first step of any workflow leading to a new journal publication starts with a formal submission of a manuscript written by someone, hereinafter the *author*. This action expresses, at the same time, interest on the topics of the journal, and may acknowledge, indirectly, the quality of the journal itself – since authors (usually) would like to publish articles in a venue that they consider respectful and qualitatively worth for different reasons (e.g., appropriateness of topic, quality of reviews, journal impact factor, definite timing of the publishing process). Then, in the next step, i.e., the reviewing phase, the person (designated by the publisher) in charge of the quality of submitted material, hereinafter the *editor*, invites other people (hereinafter the *reviewers*) to assess the quality of the submitted manuscript. The opinions returned by the reviewers are the fundamental input that the editor uses to decide upon the fate of the manuscript during the next step, i.e., the decision phase. Finally, if the manuscript has been considered worth of publication in the present form, the editor will acknowledge the author of the acceptance of his/her work – and the next steps of the workflow will be in charge of the publisher itself. Otherwise, if the article is not ready for being published, the editor either may ask for its rejection, thus terminating the workflow, or (s)he can return a list of issues to be addressed

[30] In the last two rows of the table, we indicate a mapping to a property chain, though this is not allowed in OWL2 description logic fragment, which only accepts the logical pattern: `<PropertyChain> SubPropertyOf <ObjectProperty>`, while there we would need the inverse pattern: `<ObjectProperty> SubPropertyOf <PropertyChain>`.

[31] Semantic Web Journal data: http://semantic-web-journal.com/sejp.

Table 1
A summary of the main entities of PWO which have relations to some entities introduced in the patterns used.

PWO entity	Pattern entity	Description
Workflow	Plan (*basic plan description*, via *basic plan*)	The class of descriptions (cf. *description and situation* pattern) aggregating steps that describe a conceptual workflow structure, composed by a sequence of steps
WorkflowExecution	PlanExecution (*basic plan execution*, via *basic plan*)	The class of situations "executing a workflow", i.e., aggregating entities pertaining to a real-life work, and complying to the sequence of steps in the corresponding workflow model
Step	Task (*task role*, via *control flow*)	The class describing specific tasks that form the workflow model, and that are executed by actions within particular time intervals
Action	Action (*task execution* and *plan execution*) TimeIndexedSituation (*time-indexed situation*)	The class of any (time-indexed) action with at least one agent that is participant in it and that is linked to a workflow execution by means of the property *involvesAction* and to a step of the related workflow description by means of the property *isExecutedIn* (defined in *task execution*).
executes	satisfies (*basic plan*)	The object property linking an execution of a workflow to the related workflow description
hasStep	definesTask (*basic plan description*)	An object property linking a workflow model to a component step
hasFirstStep	definesTask (*basic plan description*)	A sub-property of *hasStep* which identifies the starting step of a workflow
hasNextStep	directlyPrecedes (*sequence*)	An object property linking a step in a workflow with the step that directly follows it
hasPreviousStep	directlyFollows (*sequence*)	An object property linking a step in a workflow with the step that directly precedes it
needs	isDefinedIn **o** describes (*basic plan description*)	The object property linking a workflow step to anything required to undertake that step, e.g., an entity participating in an action of that step, or an entity participating in another action *described* (cf. the *description and situation* pattern) by the same workflow model (e.g., executing another step in that workflow)
produces	isExecutedIn (**o** hasParticipant) (*task execution* and *participation*)	The object property linking a workflow step to anything the step produces, creates or results in, e.g., an entity participating in an action of such step

to the author in order to deserve publication. In this latter case, the revision phase starts, and the author revises the paper according to reviewers' comments and editor's suggestions, and the workflow continues with a new submission phase.

The whole publishing workflow we have described (summarised in Fig. 2) can be formally represented by means of PWO. In the following excerpt (in Turtle [26]) we create an instance of the class *pwo:Workflow* as composed by a definite (we consider only the first four steps in this example) number of steps[32]:

```
:workflow a pwo:Workflow ;
  pwo:hasFirstStep :step-one ;
  pwo:hasStep
    :step-two ,
    :step-three ,
    :step-four .

:step-one a pwo:Step ; # Submission step
  pwo:hasNextStep :step-two .

:step-two a pwo:Step ; # Reviewing step
  pwo:hasNextStep :step-three ;
  parameter:hasParameter [
      a time:DurationDescription ;
      time:weeks "5"^^xsd:decimal ] .

:step-three a pwo:Step ; # Notification step
  pwo:hasNextStep :step-four .

:step-four a pwo:Step . # Revision step
```

[32]The full Turtle sources of this example are available in [22].

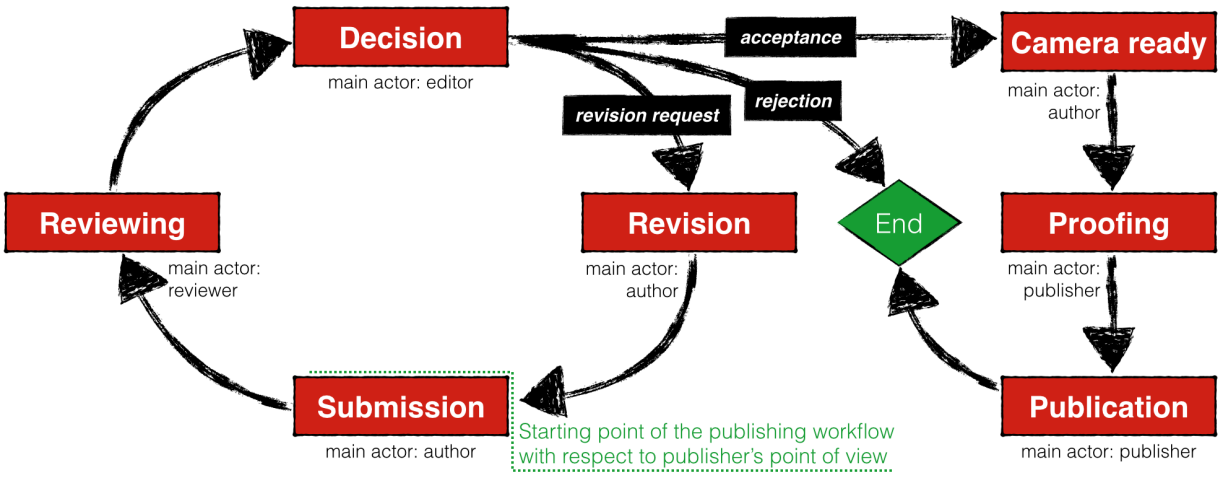

Fig. 2. A diagram describing the typical publishing workflow of a journal article – note that it does not take into account any withdrawing action by the author, nor any comment made by users on publisher's website before/after article publication.

In the next sections we show how to describe the first four steps of such workflow by taking into account real publication data available in the Semantic Web Journal Linked Data repository concerning [4].

5.1.1. Submission

The first step of the workflow concerns the submission of a manuscript by one of its authors, in this case Paolo Ciccarese. When executing this step, the manuscript receives the "submitted" status, and it is made available to the journal editor and the reviewers for the next step of the workflow. In order to describe all the aspects concerning the first step, we use several entities defined in the ontology patterns imported by PWO, as well as a number of other entities from another SPAR ontology, i.e., the *Publishing Status Ontology* (*PSO*)[33] [24]. PSO describes the status held by a document or another publication entity at each of the stages in the publishing process. In addition, existing entities of the Semantic Web Journal Linked Data repository (e.g., people and manuscripts) are reused in order to demonstrate the flexibility of PWO in working with other existing models and data, as shown below:

```
# Workflow execution
:workflow-execution a pwo:WorkflowExecution ;
  pwo:execute :workflow .

# Executing step 1: Submission
:workflow-execution
  pwo:involvesAction :submission-action .

:step-one
  taskex:isExecutedIn :submission-action ;
  pwo:needs swj-node:432 ;
  pwo:produces :submitted-status .

# The event in which one of the authors
# submits the manuscript
:submission-action a pwo:Action ;
  dcterms:description
    "Paolo Ciccarese submits the paper" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-01-21T10:08:28"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-01-21T10:08:28"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:paolo-ciccarese , swj-node:432 .

# The new status 'submitted' associated
# to the paper after the submission
:submitted-status a pso:StatusInTime ;
  pso:isStatusHeldBy swj-node:432 ;
  pso:isAcquiredAsConsequenceOf
    :submission-action ;
  pso:withStatus pso:submitted ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-01-21T10:08:28"^^xsd:dateTime ] .
```

5.1.2. Reviewing

The step regarding the reviewing phase begins with the action of the editor, Giancarlo Guizzardi in the working example, who looks for appropriate reviewers. Once reviewing requests are accepted, reviewers download the manuscript, review it, and write down their comments that are eventually sent back to the editor. In order to describe all the aspects concerning the second step, we use several entities defined in additional SPAR ontologies, i.e., the *Citation Counting*

[33] http://purl.org/spar/pso

and Context Characterisation Ontology (*C4O*)[34] [5] the *Citation Typing Ontology* (*CiTO*)[35] [23], in order to express the content of reviews and to explicitly link them to the reviewed manuscript. In the following excerpt we introduce the formalisation of the second step in PWO:

```
# Executing step 2: Reviewing
:workflow-execution pwo:involvesAction
  :choosing-reviewers-action ,
  :reviewing-action ,
  :reviews-notification-sending-action .

:step-two
  taskex:isExecutedIn
    :choosing-reviewers-action ,
    :reviewing-action ,
    :reviews-notification-sending-action ;
  # The review process can start only when
  # a manuscript has been submitted
  pwo:needs swj-node:432 , :submitted-status ;
  pwo:produces
    :review-1, :review-2,
    :under-review-status, :reviewed-status.

:choosing-reviewers-action a pwo:Action ;
  dcterms:description "The editor, Giancarlo
       Guizzardi, chooses Csaba Veres and Fernando
       Naufel do Amaral as reviewers of the
       manuscript" ;
  tisit:atTime [ a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-02-18T17:04:32"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-02-18T17:09:56"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:csaba-veres ,
    swj:fernando-naufel-do-amaral ,
    swj:giancarlo-guizzardi ,
    swj-node:432 .

:reviewing-action a pwo:Action ;
  dcterms:description "Reviewers review the
       manuscript" ;
  tisit:atTime [ a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-02-26T12:00:07"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-04-01T05:53:24"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:csaba-veres ,
    swj:fernando-naufel-do-amaral ,
    swj-node:432 .

:reviews-notification-sending-action
  a pwo:Action ;
  dcterms:description "The reviews are sent to the
       editor" ;
  tisit:atTime [ a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-03-14T11:16:34"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-04-01T05:53:24"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:csaba-veres ,
    swj:fernando-naufel-do-amaral ,
    :review-1 ,
    :review-2 ,
    swj:giancarlo-guizzardi .

# Review 1 by Csaba Veres
:review-1 a fabio:Comment ;
  frbr:realizationOf [ a fabio:Review ] ;
  cito:reviews swj-node:432 ; frbr:realizer swj:
       csaba-veres ;
  c4o:hasContent "The paper addresses a very
       practical..." .

# Review 2 by Fernando Naufel do Amaral
:review-2 a fabio:Comment ;
  frbr:realizationOf [ a fabio:Review ] ;
  cito:reviews swj-node:432 ;
  frbr:realizer swj:fernando-naufel-do-amaral ;
  c4o:hasContent "The paper presents the Collection
        Ontology (CO)..." .

# The paper has been assigned to the
# under-review status for a while
:under-review-status a pso:StatusInTime ;
  pso:isStatusHeldBy swj-node:432 ;
  pso:isAcquiredAsConsequenceOf :reviewing-action ;
  pso:isLostAsConsequenceOf
    :reviews-notification-sending-action ;
  pso:withStatus pso:under-review ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-02-26T12:00:07"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-04-01T05:53:24"^^xsd:dateTime ] .

# The paper status has changed in 'reviewed'
# after reviewers' comments
:reviewed-status a pso:StatusInTime ;
  pso:isStatusHeldBy swj-node:432 ;
  pso:isAcquiredAsConsequenceOf
    :reviews-notification-sending-action ;
  pso:withStatus pso:reviewed ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-04-01T05:53:24"^^xsd:dateTime ] .
```

5.1.3. Decision

During the third step, the editor is responsible for the fate of the paper and provides a decision for it, according to reviewers' comments. Once the decision is formalised, a decision letter is sent by email to the corresponding author (Paolo Ciccarese in the example), and the status of the paper is changed to "minor revision". In the following excerpt we introduce the formalisation of the third step in PWO:

```
# Executing step 3: Notification
:workflow-execution pwo:involvesAction
  :decision-action , :notification-action .

:step-three
  taskex:isExecutedIn
    :decision-action , :notification-action ;
  pwo:needs
    swj-node:432 , :review-1 , :review-2 ;
  pwo:produces
    :minor-revision-status , :decision-letter .

:decision-action a pwo:Action ;
  dcterms:description "The editor decides for
       acceptance or not" ;
```

[34] http://purl.org/spar/c4o
[35] http://purl.org/spar/cito

```
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-04-01T05:53:24"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-06-10T17:47:53"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:giancarlo-guizzardi ,
    :review-1 , :review-2 ,
    swj-node:432 .

:notification-action a pwo:Action ;
  dcterms:description "The editor notifies his
       decision to the corresponding author (i.e.,
       Paolo Ciccarese)." ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-06-10T17:47:53"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-06-10T17:47:53"^^xsd:dateTime ] ;
  part:hasParticipant
    swj:giancarlo-guizzardi ,
    :decision-letter ,
    :review-1 , :review-2 ,
    swj:paolo-ciccarese ,
    swj-node:432 .

# The decision letter written by the editor
:decision-letter
  a fabio:Letter , fabio:Email ;
  frbr:realizationOf [ a fabio:Opinion ] ;
  cito:citesAsRelated swj-node:432 ;
  frbr:realizer swj:giancarlo-guizzardi ;
  c4o:hasContent "Dear authors, Thank you for your
       interest in..." .

# The minor revision status assigned to the paper
     after editor's decision
:minor-revision-status a pso:StatusInTime ;
  pso:isStatusHeldBy swj-node:432 ;
  pso:isAcquiredAsConsequenceOf
    :decision-action ;
  pso:withStatus swj:minorRevision ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-06-10T17:47:53"^^xsd:dateTime ] .
```

5.1.4. Revision

During the fourth step, the authors work in order to revise the content of the previous version of the paper according to reviewers' comments and editor's suggestions. At the end of the execution of this step, the main result is the creation of a new version of the paper (e.g., *swj-node:506* in our example) that is submitted to the next step. In the following excerpt we introduce the formalisation of the fourth step of in PWO:

```
# Executing step 4: Revision
:workflow-execution pwo:involvesAction
  :revision-action .

:step-four
  taskex:isExecutedIn :revision-action ;
  pwo:needs
    swj-node:432 , :decision-letter ,
    :review-1 , :review-2 ;
  pwo:produces swj-node:506 .

:revision-action a pwo:Action ;
  dcterms:description
    "The authors revises the paper" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2013-06-10T17:47:53"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2013-07-01T05:51:30"^^xsd:dateTime ] ;
  part:hasParticipant
    swj-node:432 , :decision-letter ,
    :review-1 , :review-2 ,
    swj:silvio-peroni , swj:paolo-ciccarese .
```

5.1.5. Querying the dataset

In Section 1 we have claimed that the explicit description of publishing process data according to appropriate models, such as PWO, may result in increasing the transparency of the (publishing) process, and in enabling the use of such data for statistical analysis.

In order to show how to query the data introduced in the previous sections in view of the aforementioned goals, we introduce a typical scenario in the scholarly publishing domain. We are interested in understanding if the estimated time for completing a certain step in a workflow is enough for completing all the actual actions related to such step. As a subtask of this quite generic scenario, we could check whether the actual time needed by reviewers for reviewing a paper is within the time range originally estimated by the editors-in-chief of the journal.

Addressing such query by means of SPARQL over PWO-designed data is quite straightforward. For instance, the following SPARQL query returns all the steps described in the example, with their estimated duration (if specified), accompanied by the actual duration needed to address all the actions related to such step.

```
PREFIX ti: <http://www.ontologydesignpatterns.org/
     cp/owl/timeinterval.owl#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX pwo: <http://purl.org/spar/pwo/>
PREFIX taskex: <http://www.ontologydesignpatterns.
     org/cp/owl/taskexecution.owl#>
PREFIX tisit: <http://www.ontologydesignpatterns.
     org/cp/owl/timeindexedsituation.owl#>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX parameter:
  <http://www.ontologydesignpatterns.org/cp/owl/
       parameter.owl#>

SELECT ?execution ?step ?original_duration
       (sum(?dur_action) as ?actual_duration)
WHERE {
  {
    SELECT ?execution ?step
           (min(?dateStart) AS ?start)
           (min(?dateEnd) AS ?end)
    WHERE {
      ?step a pwo:Step ;
        taskex:isExecutedIn ?a .
```

```
      ?a tisit:atTime ?time .
      ?execution a pwo:WorkflowExecution ;
        pwo:involvesAction ?a .
      ?time ti:hasIntervalStartDate ?dateStart ;
        ti:hasIntervalEndDate ?dateEnd
    } GROUP BY ?execution ?step ?time
  }
  OPTIONAL {
    ?step parameter:hasParameter [
      time:weeks ?weeks ] .
    BIND (?weeks * 7 as ?original_duration)
  }
  BIND (day(?end - ?start) as ?dur_action)
}
GROUP BY ?execution ?step ?original_duration
ORDER BY ?start
```

Looking at the result of this query, it is straightforward to see that the second step (i.e., the reviewing phase) took fifteen days more than the estimated time, while the step that took most time to be executed was the third one (i.e., the decision phase).

5.2. An example of workflow in the legislative domain

Although PWO had been thought to describe workflows related with the scholarly publishing domain, it has been developed on purpose as an ontology for the description of generic publishing workflows. To this end, in this section we show how to use PWO to describe a workflow that concerns the process of codification of a particular law of the United States legislation, i.e., the codification of Title 41 of the United States Code, as described in an introductory webpage of the Office of the Law Revision Counsel[36].

A graphical summarisation of such codification process is shown in Figure 3[37], while a PWO exemplification of the workflow description is presented as follows:

```
# Initial workflow description
:workflow a pwo:Workflow ;
  pwo:hasFirstStep :step-one ;
  pwo:hasStep
    :step-two ,
    :step-three ,
    :step-four .

# Introduction of the codification bill
:step-one a pwo:Step ;
  pwo:hasNext :step-two .

# The bill was reported by the
# Committee Judiciary House
:step-two a pwo:Step ;
  pwo:hasNext :step-three .

# The bill was passed by
# the House of Representatives
:step-three a pwo:Step ;
  pwo:hasNext :step-four .

# The bill became Public Law 111-350
:step-four a pwo:Step .
```

5.2.1. Introduction

The first step of this workflow is the introduction of a new codification, in our example the codification bill "H.R. 1107" by Chairman Conyers and Ranking Member Smith on February 23, 2009. The result of the first step is the production of a first version of such bill. The step is rendered through PWO as follows[38]:

```
# Executing step 1: Introduction
:workflow-execution
  pwo:involvesAction :drafting-action .

:step-one
  taskex:isExecutedIn :drafting-action ;
  pwo:produces
    :hr-1107-bill-first-version .

:drafting-action a pwo:Action ;
  dcterms:description "Drafting the codification
       bill H.R. 1107" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalEndDate
      "2009-02-23T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:John_Conyers ,
    <http://dbpedia.org/resource/Adam_Smith_(
        politician)>

:hr-1107-bill a fabio:Work ;
  frbr:realization :hr-1107-bill-first-version .

:hr-1107-bill-first-version a fabio:Expression ;
  frbr:realizer
    dbpedia:John_Conyers ,
    dbpedia:Adam_Smith_(politician) .
```

5.2.2. Referring

The second step of the process concerns the establishment of a Committee, in the leading example, the Committee on the Judiciary of the House of Representatives, which referred the codification bill. In particular, H.R. 1107 was ordered favourably reported by the Committee on March 18, 2009. The step is rendered through PWO as follows:

```
# Executing step 2: Reporting by CJS
:workflow-execution pwo:involvesAction
  :referring-bill-committee-judiciary-house .

:step-two
  taskex:isExecutedIn
```

[36] Positive law codification of title 41 of the United States Code: http://uscode.house.gov/codification/t41/index.html.

[37] The full Turtle sources of this example are available in [21].

[38] The RDF representation of the agents involved in this and in the following examples are taken from DBpedia, where the prefix *dbpedia* stands for http://dbpedia.org/resource/.

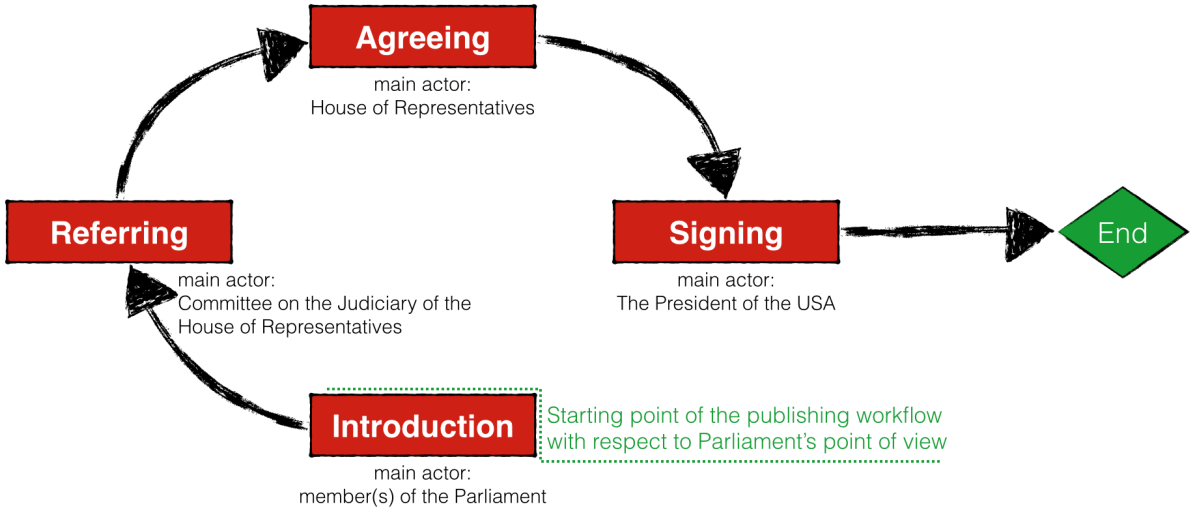

Fig. 3. A diagram exemplifying the workflow of the codification of Title 41 of the United States Code. This diagram is not intended as a complete representation of any codification process of US titles, but rather as a graphical representation of the example introduced in this section.

```
    :referring-bill-committee-judiciary-house ;
  pwo:needs :hr-1107-bill-first-version ;
  pwo:produces
    :hr-1107-bill-amended ,
    :hr-1107-bill-reported-by-committee-judiciary-
        house .

:referring-bill-committee-judiciary-house
  a pwo:Action ;
  dcterms:description
    "Referring the codification bill H.R. 1107" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-02-23T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2009-03-18T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:
        United_States_House_Committee_on_the_Judiciary
        ,
    :hr-1107-bill-first-version .

:hr-1107-bill-amended a fabio:Expression ;
  frbr:realizationOf :hr-1107-bill ;
  frbr:realizer
    dbpedia:
        United_States_House_Committee_on_the_Judiciary
        ;
  frbr:revisionOf :hr-1107-bill-first-version ;
  pso:holdsStatusInTime
    :hr-1107-bill-reported-by-committee-judiciary-
        house .

:hr-1107-bill-reported-by-committee-judiciary-house
  a pso:StatusInTime ;
  pso:isStatusHeldBy :hr-1107-bill-amended ;
  pso:withStatus :reported ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-03-18T00:00:00Z"^^xsd:dateTime ] ;
  pso:isAcquiredAsConsequenceOf
    :referring-bill-committee-judiciary-house .

:reported a pso:Status .
```

5.2.3. Agreeing

The third step of the process concerns passing a codification. In the example, the bill was passed by the House of Representatives on May 6, 2009. H.R. 1107 was referred to the Senate Judiciary Committee on May 7, 2009. H.R. 1107 with amendments was passed by the Senate by unanimous consent on December 2, 2010. Then, a message on the Senate action was sent to the House of Representatives. The House of Representatives agreed to the Senate amendments on December 17, 2010. The step is rendered through PWO as follows:

```
# Executing step 3: Agreeing
:workflow-execution pwo:involvesAction
  :passing-by-hr ,
  :receiving-bill ,
  :referring-bill-committee-judiciary-senate ,
  :sending-message-to-house-representatives .

:step-three
  taskex:isExecutedIn
    :passing-by-hr ,
    :receiving-bill ,
    :referring-bill-committee-judiciary-senate ,
    :sending-message-to-house-representatives ;
  pwo:needs
    :hr-1107-bill-amended ,
    :hr-1107-status-reported-by-committee-judiciary
        -house ;
  pwo:produces
    :hr-1107-status-passed-by-committee-judiciary-
        senate ,
    :message-on-senate-action ,
    :hr-1107-status-agreed-by-house-representatives
        .

:passing-hr a pwo:Action ;
  dcterms:description
    "Passing the codification bill H.R. 1107 by the
        House of Representatives" ;
```

```
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-05-06T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:United_States_House_of_Representatives
        ,
    :hr-1107-bill-amended .

:receiving-bill a pwo:Action ;
  dcterms:description "Receiving the codification
       bill H.R. 1107" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-05-06T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:United_States_Senate .

:hr-1107-status-passed-by-committee-judiciary-
     senate
  a pso:StatusInTime ;
  pso:isStatusHeldBy :hr-1107-bill-amended ;
  pso:withStatus :passed ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2010-12-02T00:00:00Z"^^xsd:dateTime ] ;
  pso:isAcquiredAsConsequenceOf
    :referring-bill-committee-judiciary-senate .

:referring-bill-committee-judiciary-senate
  a pwo:Action ;
  dcterms:description "Referring the codification
       bill H.R. 1107" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-05-06T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:
        United_States_Senate_Committee_on_the_Judiciary
      ,
    :hr-1107-bill-amended .

:hr-1107-status-agreed-by-committee-judiciary-
     senate
  a pso:StatusInTime ;
  pso:isStatusHeldBy :hr-1107-bill-amended ;
  pso:withStatus :passed ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2010-12-02T00:00:00Z"^^xsd:dateTime ] ;
  pso:isAcquiredAsConsequenceOf
    :referring-bill-committee-judiciary-senate .

:sending-message-to-house-representatives a pwo:
     Action ;
  dcterms:description "Sending a message to the
       House of Representatives about Senate action
        on of H.R. 1107" ;
  tisit:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2009-05-06T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:United_States_House_of_Representatives
        ,
    :hr-1107-status-passed-by-committee-judiciary-
        senate ,
    :message-to-house-representatives .

:message-on-senate-action
  a fabio:Expression ;
  frbr:realizer
    dbpedia:
        United_States_Senate_Committee_on_the_Judiciary
      ;
  frbr:realizationOf [
    a frbr:Work ;
    frbr:subject
      :hr-1107-status-passed-by-committee-judiciary
          -senate ,
      :referring-bill-committee-judiciary-senate ]
          .

:hr-1107-status-agreed-by-house-representatives
  a pso:StatusInTime ;
  pso:withStatus :agreed ;
  pso:isStatusHeldBy :hr-1107-bill-amended ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  pso:isAcquiredAsConsequenceOf
    :referring-bill-committee-judiciary-senate .

:passed a pso:Status .

:agreed a pso:Status .
```

5.2.4. Signing

In the latest step of the process the signature of the President of U.S. is required to get a codification its legal status. H.R. 1107 was signed by the President on January 4, 2011, and became Public Law 111-350, which was codified in Title 41 of the United States Code.

```
# Executing step 4: Signing
:workflow-execution pwo:involvesAction
  :hr-1107-passing-by-president .

:step-four
  taskex:isExecutedIn
    :hr-1107-passing-by-president ;
  pwo:needs
    :hr-1107-bill-amended ,
    :hr-1107-status-agreed-by-house-representatives
        ;
  pwo:produces
    :hr-1107-status-signed-by-president ,
    :public-law-111-350 ,
    :title-41-enacted-by-public-law-111-350 .

:hr-1107-passing-by-president a pwo:Action ;
  dcterms:description "Passing the codification
       bill H.R. 1107 by the President for signing
       it" ;
  tisit:atTime [ a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ;
    ti:hasIntervalEndDate
      "2011-01-04T00:00:00Z"^^xsd:dateTime ] ;
  part:hasParticipant
    dbpedia:President_of_the_United_States ,
    :hr-1107-bill-amended .
```

```
:hr-1107-status-signed-by-president
  a pso:StatusInTime ;
  pso:isStatusHeldBy :hr-1107-bill-amended ;
  pso:withStatus :signed ;
  tvc:atTime [
    a ti:TimeInterval ;
    ti:hasIntervalStartDate
      "2010-12-17T00:00:00Z"^^xsd:dateTime ] ;
  pso:isAcquiredAsConsequenceOf
    :hr-1107-passing-by-president .

:public-law-111-350 a fabio:Work ;
  frbr:adaptationOf :hr-1107-bill-amended ;
  frbr:realization [ a fabio:Expression ;
    cito:providesExcerptFor
      :title-41-enacted-by-public-law-111-350 ] .

:title-41 a fabio:Work ;
  frbr:realization
    :title-41-negative-law ,
    :title-41-enacted-by-public-law-111-350 .

:title-41-enacted-by-public-law-111-350
  a fabio:Expression ;
  frbr:revisionOf :title-41-negative-law .

:signed a pso:Status .
```

6. Conclusion

In this paper we introduced the *Publishing Workflow Ontology* (*PWO*), i.e., an OWL 2 DL ontology that is part of the Semantic Publishing and Referencing (SPAR) Ontologies [20]. PWO enables the description of publishing workflows in RDF. The whole ontology is entirely founded on existing ontology design patterns that allow us to satisfy the identified requirements by reusing best practices. We have shown two applications of PWO: describing a real publishing workflow for the publication of an article [4] in the Semantic Web Journal, and the process of codification of laws within U.S. legislation – in which we have reused entities and data from existing resources such as the Semantic Web Journal Linked Dataset.

Although PWO was designed for modelling publishing-related workflows, it can be easily reused of generic workflows, since it stands on top of abstract, off-the-shelf ontology design patterns. PWO is aligned to other workflow-related models, e.g., PROV-O, the Research Object ontology and the ontologies described in Section 2. In addition, we are studying the applicability of PWO to other domains, e.g., for describing workflows of computational or data manipulation in scientific applications. Finally, we also plan to develop some APIs in order to facilitate the adoption of PWO within existing publishing platforms.

References

[1] Belhajjame, K., Corcho, O., Garijo, D., Zhao, J., Missier, P., Newman, D. R., Palma, R., Bechhofer, S., García, E., Gómez-Pérez, J. M., Klyne, G., Ruiz, J. E., Soiland-Reyes, S., De Roure, D., & Goble, C. (2012). Workflow-Centric Research Objects: A First Class Citizen in the Scholarly Discourse. In Garcia Castro, A., Lange, C., van Harmelen, F., Good, B. (Eds.), Proceedings of the 2nd Workshop on Semantic Publishing (SePublica 2012). Aachen, Germany: CEUR-WS.org. http://ceur-ws.org/Vol-903/paper-01.pdf

[2] Belhajjame, K., Zhao, J., Garijo, D., Hettne, K. M., Palma, R., Corcho, O., Gómez-Pérez, J. M., Bechhofer, S., Klyne, G., & Goble, C. A. (2014). The Research Object Suite of Ontologies: Sharing and Exchanging Research Data and Methods on the Open Web. The Computing Research Repository (CoRR), abs/1401.4307. http://arxiv.org/abs/1401.4307

[3] Blomqvist, E., Gangemi, A., Daga, E., & Presutti, V. (2010). Experimenting with eXtreme Design. In Cimiano, P., Pinto, H. S. (Eds.), Proceedings of the Conference on Knowledge Engineering and Knowledge Management (EKAW 2010): 120–134. Heidelberg, Germany: Springer. http://dx.doi.org/10.1007/978-3-642-16438-5_9

[4] Ciccarese, P., & Peroni, S. (2014). The Collections Ontology: creating and handling collections in OWL 2 DL frameworks. Semantic Web, 5(6): 515–529. http://dx.doi.org/10.3233/SW-130121

[5] Di Iorio, A., Nuzzolese, A. G., Peroni, S., Shotton, D., & Vitali, F. (2014). Describing bibliographic references in RDF. In Garcia Castro, A., Lange, C., Lord, P., Stevens, R. (Eds.), Proceedings of 4th Workshop on Semantic Publishing (SePublica 2014). Aachen, Germany: CEUR-WS.org. http://ceur-ws.org/Vol-1155/paper-05.pdf

[6] Falco, R., Gangemi, A., Peroni, S., & Vitali, F. (2014). Modelling OWL ontologies with Graffoo. In Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (Eds.), ESWC 2014 Satellite Events - Revised Selected Papers: 320–325. Cham, Switzerland: Springer. http://dx.doi.org/10.1007/978-3-319-11955-7_42

[7] Gangemi, A. (2008). Norms and plans as unification criteria for social collectives. Autonomous Agents and Multi-Agent Systems, 17 (1): 70–112. http://dx.doi.org/10.1007/s10458-008-9038-9

[8] Gangemi, A., Borgo, S., Catenacci, C., & Lehmann, J. (2004). Task taxonomies for knowledge content. METOKIS Deliverable D7. http://metokis.salzburgresearch.at/files/deliverables/metokis_d07_task_taxonomies _final.pdf

[9] Gangemi A., & Presutti V. (2009). Ontology Design Patterns. In Staab, S., Studer, R. (Eds.), Handbook of Ontologies (2nd edition): 221–243. Heidelberg, Germany: Springer. http://dx.doi.org/10.1007/978-3-540-92673-3_10

[10] Gangemi, A., Peroni, S., Shotton, D., & Vitali, F. (2014). A pattern-based ontology for describing publishing workflows. In de Boer, V., Gangemi, A., Janowicz, K., Lawrynowicz, A. (Eds.), Proceedings of the 5th International Workshop on Ontology and Semantic Web Patterns (WOP 2014), CEUR Workshop Proceedings 1302: 2–13. Aachen, Germany: CEUR-WS.org. http://ceur-ws.org/Vol-1302/paper1.pdf

[11] Garijo, D., & Gil, Y. (2011). A new approach for publishing workflows: abstractions, standards, and linked data. In Pro-

ceedings of the 6th workshop on Workflows in support of large-scale science (WORKS 2011): 47–56. New York, New York, USA: ACM. http://dx.doi.org/10.1145/2110497.2110504

[12] Hettne, K., Soiland-Reyes, S., Klyne, G., Belhajjame, K., Gamble, M., Bechhofer, S., Roos, M., & Corcho, O. (2012). Workflow forever: semantic web semantic models and tools for preserving and digitally publishing computational experiments. In Proceedings of the 4th International Workshop on Semantic Web Applications and Tools for the Life Sciences (SWAT4LS 2011): 36–37. http://dx.doi.org/10.1145/2166896.2166909

[13] Horridge, M., & Patel-Schneider, P. F. (2012). OWL 2 Web Ontology Language: Manchester Syntax (Second Edition). W3C Working Group Note, 11 December 2012. http://www.w3.org/TR/owl2-manchester-syntax/

[14] Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission, 21 May 2004. http://www.w3.org/Submission/SWRL/

[15] Hu, Y., Janowicz, K., McKenzie, G., Sengupta, K., & Hitzler, P. (2013). A Linked-Data-Driven and Semantically-Enabled Journal Portal for Scientometrics. In Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J. X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (Eds.), Proceedings of the 12th International Semantic Web Conference (ISWC 2013): 114–129. Heidelberg, Germany: Springer. http://dx.doi.org/10.1007/978-3-642-41338-4_8

[16] Lebo, T., Sahoo, S., & McGuinness, D. (2013). PROV-O: The PROV Ontology. W3C Recommendation, 30 April 2013. http://www.w3.org/TR/prov-o/

[17] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., & Bizer, C. (2015). DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Semantic Web, 6 (2): 167–195. http://dx.doi.org/10.3233/SW-140134

[18] Noy, N. F., Chugh, A., Liu, W., & Musen, M. A. (2006). A Framework for Ontology Evolution in Collaborative Environments. In Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (Eds.), Proceedings of the 5th International Semantic Web Conference (ISWC 2006): 544–558. Heidelberg, Germany: Springer. http://dx.doi.org/10.1007/11926078_39

[19] O'Connor, C., Haenel, S., Gnanapiragasam, A., Hepp, M., & Fleischer, T. (2015). Building an Automated XML-Based Journal Production Workflow. In Proceeding of the Journal Article Tag Suite Conference 2015 (JATS-Con 2015). Bethesda, Maryland, USA: National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov/books/NBK279927/

[20] Peroni, S. (2014). The Semantic Publishing and Referencing Ontologies. In Semantic Web Technologies and Legal Scholarly Publishing: 121–193. Cham, Switzerland: Springer. http://dx.doi.org/10.1007/978-3-319-04777-5_5

[21] Peroni, S. (2015). Example of use of PWO: legislative domain. figshare. http://dx.doi.org/10.6084/m9.figshare.1449044

[22] Peroni, S. (2015). Example of use of PWO: publishing domain. figshare. http://dx.doi.org/10.6084/m9.figshare.1449043

[23] Peroni, S., & Shotton, D. (2012). FaBiO and CiTO: Ontologies for describing bibliographic resources and citations. Web Semantics, 17: 33–43. http://dx.doi.org/10.1016/j.websem.2012.08.001

[24] Peroni, S., Shotton, D., & Vitali, F. (2012). Scholarly publishing and linked data: describing roles, statuses, temporal and contextual extents. In Proceedings of the 8th International Conference on Semantic Systems (i-Semantics 2012): 9–16. http://dx.doi.org/10.1145/2362499.2362502

[25] Presutti, V., Gangemi, A., David, S., Aguado, G., Suárez-Figueroa, M. C., Montiel-Ponsoda, E., & Poveda, M. (2008). A Library of Ontology Design Patterns. NeOn deliverable D2.5.1. http://neon-project.org/deliverables/WP2/NeOn_2008_D2.5.1.pdf

[26] Prud'hommeaux, E., & Carothers, G. (2014). Turtle - Terse RDF Triple Language. W3C Recommendation, 25 February 2014. http://www.w3.org/TR/turtle/

[27] Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., & Mulyar, N. (2006). Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22. http://www.workflowpatterns.com/documentation/documents/BPM-06-22.pdf

[28] Sebastian, A., Noy, N. F., Tudorache, T., & Musen, M. A. (2008). A Generic Ontology for Collaborative Ontology-Development Workflows. In Gangemi, A., Euzenat, J. (Eds.), Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008): 318–328. http://dx.doi.org/10.1007/978-3-540-87696-0_28

[29] Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., Soiland-Reyes, S., Dunlop, I., Nenadic, A., Fisher, P., Bhagat, J., Belhajjame, K., Bacall, F., Hardisty, A., de la Hidalga, A. N., Vargas, M. P. B., & Goble, C. (2013). The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. Nucleic Acids Research, 41 (W1): W557–W561. http://dx.doi.org/10.1093/nar/gkt328.