Christophe Debruyne, Jonathan Riggio, Olga De Troyer, and Declan O'Sullivan. An ontology for representing and annotating data flows to facilitate compliance verification. In Manuel Kolp, Jean Vanderdonckt, Monique Snoeck, and Yves Wautelet, editors, 13th International Conference on Research Challenges in Information Science, RCIS 2019, Brussels, Belgium, May 29-31, 2019, pages 1–6. IEEE, 2019

An Ontology for Representing and Annotating Data Flows to Facilitate Compliance Verification

Christophe Debruyne[1,2], Jonathan Riggio[2], Olga De Troyer[2], Declan O'Sullivan[1]
[1] Trinity College Dublin, Dublin, Ireland
[2] Vrije Universiteit Brussel, Brussels, Belgium
debruync@tcd.ie, jriggio@vub.be, olga.detroyer@vub.be, declan.osullivan@tcd.ie

***Abstract*— In this paper, we argue that there is a gap to be bridged between the development and maintenance of services and the various internal and external policies that emerge and evolve outside of these systems. To bridge this gap, we propose a semantic model, i.e. ontology, for representing Data Flows and linking them with structured representations of the data that is processed (datasets, databases, queries, etc.). Data Flow Diagramming is a technique for capturing the various data and information flows between an information system and external stakeholders as well as within such a system. This technique is used in the analysis phase of information systems development and captures the inputs and outputs of various processes. Our model allows these data flows to be presented and linked with structured representations of the data that is to be used, consulted, processed, etc. We demonstrate that this model can facilitate compliance verification processes of (intelligent) systems by allowing these flows to be analyzed. Next to the ontology, which has been made available according to best practices in the field, we furthermore posit our contributions within the state of the art.**

Keywords-component; Data Flows, Data Flow Diagrams, Semantic Models

I. INTRODUCTION

Data processing, in general, is increasingly the subject of various regulations such as the General Data Protection Regulation (GDPR)[1], an EU law that came into effect on May 2018. While not new, principles such as *privacy by design* or *data protection by design* have been explicitly mentioned by some of these legislations, encouraging developers to account for, at the start, the measures necessary to be compliant. Policies may be internal or external and even evolve over time. This means that not only the development but also the maintenance of information systems and their intelligent processes should co-evolve.

In this context, the adoption of Data Flow Diagrams to support this co-evolution appears as a practical solution as it is already widely used in industry for structured software analysis and design, providing efficient means of communication between developers and business administration [1]. However, as Data Flow Diagrams are merely used during analysis and for communication and documentation, there is a disconnection between the data flow models and the artifacts that have been produced during development phases. In order to tie the analysis and design of services with the resulting or used artifacts (databases, datasets, etc.), appropriate solutions are needed. These solutions could also facilitate data governance. Data governance is defined as *"[referring] to what decisions must be made to ensure effective management and use of IT (decision domains) and who makes the decisions (locus of accountability for decision-making)."* [2] As a solution, we propose an ontology usable for capturing data flows and annotate them with information that will allow compliance to be analyzed. The resulting model cannot only be used at design time but at any time components of the system are to evolve.

The literature on the adoption of Semantic Web technologies (i.e., models) for analyzing GDPR compliance has been investigated. Much of the work in the state of the art addresses assessing compliance *a posteriori* [3] [4], in other words: assessing the compliance of past events by, for instance, annotated and querying logs. Others propose models for describing policies and prescriptions of the various data processing steps [5], [6]. We instead propose a model to bridge the gap between both the analysis and design, and the development and maintenance of information systems, and the various policies those systems should comply with. This allows analysts to better describe systems, more in particular in terms of the (types of) data used by their processes. Therefore, it enables to verify compliance with the necessary measures that need to be put in place during the analysis and design phases of a project.

Our solution relies on standardized Semantic Web technologies – the Web Ontology Language (OWL) [7] and the Resource Description Framework (RDF) [8] – to facilitate the integration of other models, ontologies and vocabularies.

The remainder of this paper is structured as follows: Section 2 provides a brief introduction to Data Flow Diagrams (DFDs); Section 3 presents our Data Flow ontology and exemplifies its use; Section 4 illustrates how one can avail vocabularies to enrich data flows for compliance analysis, demonstrating that our model is apt to tackle aforementioned gap between the analysis and design of services, and development; in Section 5 we present the related work and argue for the use of DFDs over workflow-oriented models; and finally, in Section 6, we summarize contributions and describe future work.

[1] https://eur-lex.europa.eu/eli/reg/2016/679/oj

II. Data Flow Diagrams (DFD)

Data Flow Diagrams "*enable [one] to model how data flow through an information system, the relationships among the data flows, and how data come to be stored at specific locations. Data flow diagrams also show the processes that change or transform data.*" [9] Figure 1 depicts the various elements of a component diagram:

- *Interfaces* are rectangles and represent the external entities or "actors" that engage with the system. They can send and receive data flows. In Figure 1, interfaces are the customer, the restaurant's kitchen and the restaurant's manager.
- Processes, shown as ellipses, process the data they receive as input and send output on to other DFD entities. A process always has at least one input and at least one output. In the example, there are processes for ordering food, updating the records and inventory, and producing management reports.
- Data stores, represented as rectangles with only a top and a bottom border, represent the places where data is stored (in any form) for other processes to consult. In Figure 1, there are two data stores that facilitate keeping track of the restaurant's sales and inventory.
- *Data flows* are depicted as arrows and indicate data moving from one DFD entity to another. For example, in Figure 1, there is a data flow "order" that denotes the information submitted by the customer to the "Order Food" process.

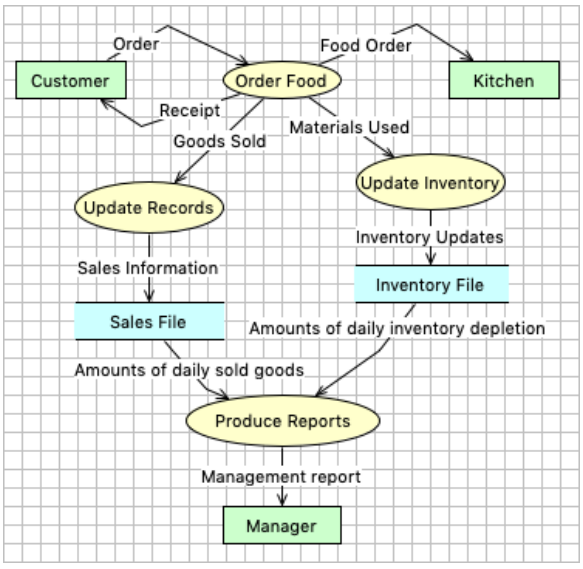

Figure 1 Example of a Data Flow Diagram (inspired from [9])

The notation also has a set of rules including that interfaces and data stores should not be connected (a process needs to be modeled in the middle); data flows should have unique names; etc. Some notations allow processes to be decomposed into subprocesses and comes with its own set of rules such as balancing: "*The conservation of inputs and outputs to a DFD process when that process is decomposed.*" [9]

DFDs are not well suited for modeling a highly distributed system as services provided by other systems should be modeled as interfaces (i.e., an "actor" outside the information system) in a DFD. As DFDs are focused on data flows, however, one can consider the distributed system as one information system. This allows one to model the services as processes (owned by actors), even though the details of which may not be known to the modelers.

For the purpose of this paper, we limit ourselves to "simple" DFD diagrams and data flows that cannot be forked or merged. Our semantic model, which we will present in the next section, can be easily extended to support more complex DFDs.

III. The DFD Ontology

This section presents an ontology for representing Data Flows. This ontology provides predicates to represent data flows and entities. We have decided not to include capturing the graphical representation of the *diagram*, as this does not pertain to the aim of analyzing the processes and flows. If this would be needed, one can easily add statements about the location, size, etc. of the various entities and flows with, for instance, WKT[2].

The ontology[3] is implemented using OWL 2 [7]. The ontology has been made available with a CC-BY-4.0 license and published according to best practices and guidelines in the Semantic Web community. Documentation was generated using WIDOCO [10], which integrated LODE [11] and WebVOWL [12]. The former generated documentation for the OWL ontology using its axioms and annotations, and the latter visualizes the ontology.

The ontology contains 4 disjoint classes: `dfd:DataFlow`, `dfd:DataStore`, `dfd:Interface`, and `dfd:Process`. `dfd:DataStore`, `dfd:Interface`, and `dfd:Process` are subclasses of `dfd:Entity`. There are two object properties relating Data Flows to DFD Entities: `dfd:from` and `dfd:to`. Both have `dfd:DataFlow` as the domain and `dfd:Entity` as range. All the components of a DFD diagram are named, for which we will reuse the `rdfs:label` predicate. All DFD entities and all data flows are supposed to have unique names in a diagram. Using Semantic Web technologies, this is a rule that is more difficult to impose. We, therefore, redefine this rule to state that all DFD entities and all data flows have a unique name and that these names are to be unique in the base or in the RDF graph they are used.

Given the DFD example of Figure 1, a snippet of the DFD's representation in RDF is given below. In this snippet, the default namespace is http://example.org/ and URI encoding

[2] WKT stands for Well-Known Text and is a markup language for representing geometric objects (https://en.wikipedia.org/wiki/Well-known_text)
[3] https://w3id.org/dfd

was used for the URIs. The RDF was generated using an Eclipse plugin written for this study (see Figure 2).

```
@base <http://example.org/> .
@prefix : <http://example.org/> .
@prefix dfd: <https://w3id.org/dfd#> .

:Customer  a          dfd:Interface ;
           rdfs:label "Customer" .
<http://example.org/Order+Food>
           a          dfd:Process ;
           rdfs:label "Order Food" .
:Order     a          dfd:DataFlow ;
           rdfs:label "Order" ;
           dfd:from   :Customer ;
           dfd:to     <http://example.org/Order+Food> .
```

Listing 1 An RDF representation of the "Order" data flow between "Customer" (an interface) and "Order Food" (a process). Note that the RDFS namespace was omitted for brevity.

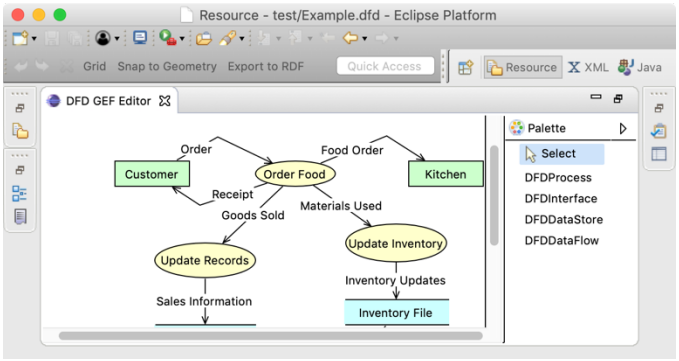

Figure 2 Graphical DFD editor written in Eclipse

IV. Knowledge Organization: Towards Annotating DFDs for Compliance Analysis

The RDF data model allows one to easily add statements to a graph. Using the RDF that was generated (as explained in the previous section), one can extend the graph with 1) structured information about the data shared or stored, and 2) the purpose of data being processed. For both, we can avail of existing (often standardized) vocabularies.

To describe data that is used in a data flow or stored in a data store, we can:

- Use a reference to a vocabulary or ontology that can be used to describe the concepts and relations that will be used in a data flow or store.
- Avail of SPARQL Inferencing Notation (SPIN) [13] to represent SPARQL queries in a structured manner. Such structured descriptions of queries are suitable for describing flows from a data store to a process. Related work has shown that these representations can be used to interrogate these queries [14].
- Describe datasets with the RDF Data Cube Vocabulary [15], which also offers a vocabulary to prescribe a data sets' "schema". The VoID Vocabulary [16] allows one to describe Linked Data datasets and offers predicates for detailing where one can find the data, such as a SPARQL endpoint.

For capturing data processing purposes, we can adopt:

- The PAM ontology proposed in [17] relating *purpose, dataset* and *consent*, with consent information limited to that what is stored by a system. Instances of `dfd:Process` can be directly linked to instances of `pam:Purpose`.
- An ontological "design pattern", recently proposed in [6], for capturing the personal data captured in an organization's privacy policy. This ontology provides predicates not only to describe the purposes for data processing, it also provides predicates for describing what data is collected, who it will be shared with, and how long it is retained for (amongst others). Combined with the ontology proposed in [5], one can interrogate the legal basis of a particular data processing activity.

It is clear that possibilities are vast, but the goal of this section is not to propose a governance framework in which all models are unified, which would be the subject of future work, but rather to demonstrate what is feasible.

A. An Example

We will use the simple example of using names and email addresses to send out newsletters. A *part* of *a* DFD capturing this process and its corresponding RDF are shown in Figure 3 and Listing 2. Note that the data flow containing the email to the customers (the interface, not the data store) has been omitted for brevity.

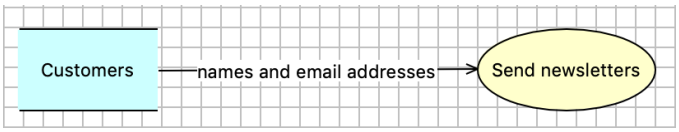

Figure 3 DFD for the process of sending newsletters

```
:Customers  a          dfd:DataStore ;
  rdfs:label  "Customers" .

<http://example.org/Send+newsletters>
  a           dfd:Process ;
  rdfs:label  "Send newsletters" .

<http://example.org/names+and+email+addresses>
  a           dfd:DataFlow ;
  rdfs:label  "names and email addresses" ;
  dfd:from    :Customers ;
  dfd:to      <http://example.org/Send+newsletters> .
```

Listing 2 An RDF representation of the DFD in Figure 3

Let's assume that the customer database is accessible via a SPARQL endpoint using FOAF[4]. The data flow can be anno-

[4] http://xmlns.com/foaf/0.1/

tated with the SPARQL query for retrieving the `foaf:name` and `foaf:mbox` of customers. We will use SPIN to represent the SPARQL query and the predicate `dfd:representedBy` to relate the data flow and the query. This would look as follows:

```
# PREFIX foaf: <http://xmlns.com/foaf/0.1/>
# SELECT ?name ?email WHERE {
#   ?x foaf:name ?name .
#   ?x foaf:mbox ?email .
# }
<http://example.org/names+and+email+addresses>
  dfd:representedBy [
        a sp:Select ;
        sp:resultVariables ( _:b1 _:b2 ) ;
        sp:where ( [ sp:object sp:_arg1 ;
                     sp:predicate foaf:name ;
                     sp:subject _:b1 ]
                   [ sp:object sp:_arg1 ;
                     sp:predicate foaf:mbox ;
                     sp:subject _:b2 ] )
  ].
_:b1  sp:varName "name"^^xsd:string .
_:b2  sp:varName "email"^^xsd:string .
```

Listing 3 The SPARQL query and its corresponding SPIN representation. Notice how the URI of the data flow is connected to the query.

The GDPROV ontology[5] proposed in [5] allows one to state RDF resources to be an instance of `gdprov:PersonalData`. We will use this ontology to state the two FOAF properties we use are such instances using `rdf:type`. Assuming that one can query the RDF representation of the data flows, its representation of the data, and information about its nature, the following SPARQL query allows an agent to find out whether a data flow is using personal data:

```
1.  SELECT ?dataflow ?predicate WHERE {
2.    ?dataflow a dfd:DataFlow .
3.    ?dataflow sp:where/rdf:rest*/rdf:first
4.                [ sp:object ?object ;
5.                  sp:predicate ?predicate ;
6.                  sp:subject ?subject ] .
7.    ?predicate a gdprov:PersonalData .
8.  }
```

Listing 4 SPARQL query to list all personal data that data flows use. Note that the third line allows us to iterate over each of the triple patterns in the where clause. The `rdf:rest*` in the property path will match with both the 0th and 1st occurrence of `rdf:rest` in our list.

The SPARQL query in Listing 4 only checks whether the predicates used are considering personal data. The query can be extended to check whether this holds true for other parts of the triple patterns. One example is to test whether the subject is an instance of personal data when the predicate is `rdf:type`. This shows that a set of SPARQL queries can be written to inform users of aspects to be taken into account.

[5] http://purl.org/adaptcentre/openscience/ontologies/gdprov#

B. *Towards governance and a feedback loop*

We demonstrated how enriched RDF representations of data flows can be used to engage with the services being designed. Those representations, when *combined* with the work, for example, proposed by [5], allows one to ask more complex questions such as: "What is the legal basis of using personal data for this process?" or "Are we allowed to use the personal data in an information purpose for a particular process?" These are the questions that appropriate data governance platforms need to answer. These models, when stored, can be furthermore used for reassessing processes whenever datasets, policies, etc. change. The models we propose have the potential to *facilitate* not only compliance analysis but also the impact of certain changes.

We emphasize that this facilitation will be achieved through integration with work proposed in for instance [3]–[5] (which includes support for formulating compliance), as the ontology is meant to bridge the gap between design, development, and monitoring. The integration of our approach and the models we adopted into a governance framework will be the subject of future work.

V. Related Work

We briefly described related work on compliance analysis in Section I. In this section, we will describe the related work on representing processes with Business Process Model and Notation (BPMN) [18] and other initiatives for representing processes with ontologies and motivate our choice for DFDs.

A. *DFD vs. BPMN*

Before discussing the reasons for adopting DFDs, we first want to elaborate on the difference between Data Flow Modeling, and Business Process Modeling and Flow Charts.

As stated in Section II, Data Flow Diagrams depict the movement of data as data flows between external entities (called interfaces), and internal entities (processes and data stores) within an information system. Data Flow Diagramming is useful to describe a solution during the *analysis* phase of the systems development life cycle (SDLC), and is not a detailed design for it. As [19] noted, DFDs focus on data and have no means to model decisions, branches, workflows, etc. DFDs thus provide a "snapshot" of all possible data flows within a system.

The Business Process Model and Notation, or BPMN [18], does provide constructs for modeling sequences with evens support for branching and converging sequences (based on decisions). The notation is used to graphically model business processes. Similar to flowcharts and UML Activity Diagrams, the models are *process-oriented* with support for events and capturing the sequence of tasks. Indeed, the major difference between the arrows in DFD and BPMN models is that the arrows in the former denote the data being exchanged and the latter the order of tasks being executed. As BPMN proposes a

logical breakdown of processes, they are more suitable for the *design* phase of the SDLC.

In [20], the authors proposed sBPMN, an ontology to serialize BMPN. Much like the purpose of our study, their aim was to enrich models, in their case with Semantic Web Services. While we are aware of these ontologies, we have adopted DFDs for two reasons. First, DFDs are used primarily during the analysis phase; they primarily focus on the data flows and provide a more abstract view of the process that a system should support. Secondly, we deem the data-centric perspective on a system more suitable for our approach; we are concerned with the data used by processes at any given time rather than a logical breakdown of processes that use that data. In DFDs, we are able to annotate both the data being exchanged (the data flows are explicit) as well as the data stores.

Although BPMN does provide support for so-called data objects and data store references (which can be associated with tasks in processes), the modeling focus is on the processes and less on the data. In addition, their representation clutters and complicates the model. For instance, when one process' output is another process' input, there is a need for additional arrows pointing to and from the data object to indicate input and output. This is demonstrated in Figure 4. This makes the models less suitable for analysis, as modeling at the same time processes and data is cognitively harder.

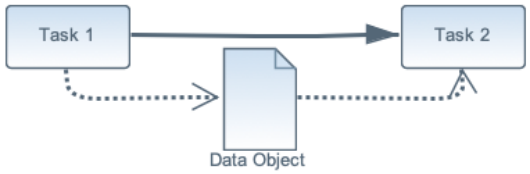

Figure 4 Tasks where one's output is the other's input in BPMN

Therefore, given the purpose of this study – representing and annotating data and information flows – we deem DFD diagrams more appropriate. However, the types of annotations proposed in this study can also be applied to BPMN models.

B. *Relation with other Vocabularies*

We already discussed the difference between our model and sBPMN in the previous section. In this section, we will discuss existing vocabularies. The vocabularies in this section all share some notion of "process", "activity" or "task".

A popular vocabulary for describing activities is provided by the Provenance Ontology PROV-O [21]. PROV-O allows one to represent provenance information in terms of *activities*, entities and agents. It is important to note that PROV-O is used to capture the provenance information of things that have happened in the past, that is why quite a few predicates are written in a past tense (e.g., `prov:wasGeneratedBy`). While PROV-O had a predicate to relate activities to a plan, it did not allow for plans to be described. This was provided by P-PLAN [22], which extended `prov:Plan` to be related with steps, which in turn correspond with activities. Both PROV-O and P-PLAN where then reused by OPMW [23], which allowed one to create workflow-templates, and instantiations thereof, of scientific processes (publishing an article, generating results, etc.). Not only is OPMW domain specific, the models that one can create with this ontology focuses on *workflows,* much like BPMN. We do note that the ontology proposed in this paper can be aligned with PROV-O when there would be a need to create instances of `dfd:Process` that are also instances of `prov:Activity`, which may be an opportunity in the future.

Furthermore, in bioinformatics an ontology for detailing workflows in that domain has been proposed [24]. [25] proposed an ontology for modeling ontology engineering workflows in the Protégé [6] ontology development environment. While generic enough for various ontology engineering projects and methodologies, it is fit for a specific type of project only. In [26], the authors proposed semantically annotating the manipulation and analysis of data in data processing "pipelines". Their efforts are much closer to the execution level of a particular data processing purpose. For this reason, we deem their contribution complementary, as the "implementation" of processes happen after the design process.

We can conclude that – to the best of our knowledge – semantic models to describe processes are not manifold, often focus on the workflow, and are at times domain specific or close to an execution model. We have, while looking into related work, not taken into account initiatives concerned with projects such as DOAP[7], as the constructs they provide are too superficial.

VI. Conclusions and Future Work

The increasing pressure for organizations to be compliant with various regulations and policies provided the motivation for this study. As organizations need to demonstrate that their data processing activities (which evolve over time) are compliant (e.g., with GDPR), they can benefit from semi-automated processes that facilitate compliance processes.

In this study, we argued that there is a disconnection between appropriate techniques for analyzing services, such as Data Flow Diagrams, and the development and maintenance of these services. We therefore proposed an ontology for representing Data Flow Diagrams, of which its instantiations can – thanks to the RDF model – be extended with structured descriptions of information (queries, datasets, databases, etc.) shared between Data Flow entities. We believe that this approach brings us closer to semi-automated compliance analysis at design time. The contributions of this work can be integrated in data governance frameworks where the roles and responsibilities of stakeholders are stored, which is the subject of future work.

[6] https://protege.stanford.edu/

[7] Description of a Project (DOAP) vocabulary: http://usefulinc.com/ns/doap

A current limitation of our ontology is the lack of support for functional decompositions of processes. While not necessary, such function decomposition does allow service designer to separate concerns and keep diagrams tidy. To support functional decomposition the ontology needs to be extended. The extension would require relations between processes and its children as well as set of constraints to validate models. As checking balanced DFDs is outside the capabilities of OWL, these will have to be written in a constraint language such as SHACL [27]. Finally, we foresee aligning our model with workflow-oriented models by integrated existing work, such as sBPMN [20], as to provide support for both analysis and design phases in the systems development life cycle.

ACKNOWLEDGEMENTS

The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund. We would also like to thank the anonymous reviewers for their valuable comments.

VII. REFERENCES

[1] Q. Li and Y.-L. Chen, "Data Flow Diagram," in *Modeling and Analysis of Enterprise and Information Systems*, Springer, 2009, pp. 85–97.

[2] V. Khatri and C. V. Brown, "Designing data governance," *Commun. ACM*, vol. 53, no. 1, pp. 148–152, 2010.

[3] P. Westphal, J. D. Fernández, S. Kirrane, and J. Lehmann, "SPIRIT: A Semantic Transparency and Compliance Stack," in *Proceedings of the Posters and Demos Track of the 14th International Conference on Semantic Systems co-located with the 14th International Conference on Semantic Systems (SEMANTiCS 2018), Vienna, Austria, September 10-13, 2018.*, 2018, vol. 2198.

[4] S. Kirrane *et al.*, "A Scalable Consent, Transparency and Compliance Architecture," in *The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers*, 2018, vol. 11155, pp. 131–136.

[5] H. J. Pandit, D. O'Sullivan, and D. Lewis, "Queryable Provenance Metadata For GDPR Compliance," in *Proceedings of the 14th International Conference on Semantic Systems (SEMANTiCS 2018), Vienna, Austria, Sep. 10-13, 2018*, 2018, pp. 262–268.

[6] H. J. Pandit, D. O'Sullivan, and D. Lewis, "An Ontology Design Pattern for Describing Personal Data in Privacy Policies," in *Proceedings of the 9th Workshop on Ontology Design and Patterns (WOP 2018) co-located with 17th International Semantic Web Conference (ISWC 2018)*, 2018.

[7] W3C OWL Working Group, "OWL 2 Web Ontology Language: Document Overview," *W3C Recommendation*, 2012. [Online]. Available: https://www.w3.org/TR/owl2-overview/.

[8] F. Manola and E. Miller, "RDF Primer," *W3C Recomm.*, 2004.

[9] J. A. Hoffer, J. F. George, and J. S. Valacich, *Modern Systems Analysis and Design*, 6th ed. Pearson, 2002.

[10] D. Garijo, "WIDOCO: A wizard for documenting ontologies," in *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, 2017, pp. 94–102.

[11] S. Peroni, D. Shotton, and F. Vitali, "The live OWL documentation environment: A tool for the automatic generation of ontology documentation," in *Knowledge Engineering and Knowledge Management - 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. Proceedings*, 2012, pp. 398–412.

[12] S. Lohmann, V. Link, E. Marbach, and S. Negru, "WebVOWL: Web-based visualization of ontologies," in *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th - to - 12th, 2018*, 2015.

[13] H. Knublauch, J. A. Hendler, and K. Idehen, "SPIN - Overview and Motivation," *http://www.w3.org/Submission/spin-overview/*, 2011. [Online]. Available: https://www.w3.org/Submission/spin-overview/.

[14] A. Meehan, D. Kontokostas, M. Freudenberg, R. Brennan, and D. O'Sullivan, "Validating interlinks between linked data datasets with the SUMMR methodology," in *On the Move to Meaningful Internet Systems: OTM 2016 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings*, 2016, pp. 654–672.

[15] R. Cyganiak and D. Reynolds, "The RDF Data Cube Vocabulary," 2014. [Online]. Available: https://www.w3.org/TR/vocab-data-cube/.

[16] K. Alexander, R. Cyganiak, M. Hausenblas, and Z. Jun, "Describing Linked Datasets with the VoID Vocabulary," *W3C Interes. Gr. Note 03 March 2011*, 2011.

[17] C. Debruyne, H. J. Pandit, D. Lewis, and D. O'Sullivan, "A Consent-aware Mapping Engine for Generating Policy-compliant Datasets," in *Proceedings - 12th IEEE International Conference on Semantic Computing, ICSC 2019*, 2019, pp. 199–203.

[18] Object Management Group, R. Parida, and S. Mahapatra, "Business Process Model and Notation (BPMN) Version 2.0," *Business*, 2011. .

[19] G. M. Giaglis, "A taxonomy of business process modeling and information systems modeling techniques," *Int. J. Flex. Manuf. Syst.*, vol. 13, no. 2, pp. 209–228, 2001.

[20] A. Witold, A. Filipowska, M. Kaczmarek, and T. Kaczmarek, "Semantically Enhanced Business Process Modeling Notation," in *Semantic Technologies for Business and Information Systems Engineering: Concepts and Applications*, S. Smolnik, F. Teuteberg, and O. Thomas, Eds. Hershey, PA: IGI Global, 2012, pp. 259–275.

[21] T. Lebo, D. McGuinness, and S. Sahoo, "PROV-O: The PROV Ontology," 2013. [Online]. Available: https://www.w3.org/TR/prov-o/.

[22] D. Garijo and Y. Gil, "The P-PLAN Ontology," 2014. [Online]. Available: http://vocab.linkeddata.es/p-plan/.

[23] D. Garijo and Y. Gil, "The OPMW-PROV Ontology," 2014. [Online]. Available: http://www.opmw.org/model/OPMW/.

[24] J. Ison *et al.*, "EDAM: An ontology of bioinformatics operations, types of data and identifiers, topics and formats," *Bioinformatics*, vol. 29, no. 10, pp. 1325–1332, 2013.

[25] A. Sebastian, N. F. Noy, T. Tudorache, and M. A. Musen, "A generic ontology for collaborative ontology-development workflows," in *International Conference on Knowledge Engineering and Knowledge Management*, 2008, pp. 318–328.

[26] M.-Á. Sicilia, E. García-Barriocanal, S. Sánchez-Alonso, M. Mora-Cantallops, and J.-J. Cuadrado, "Ontologies for Data Science: On Its Application to Data Pipelines," in *Metadata and Semantic Research*, 2019, pp. 169–180.

[27] H. Knublauch and D. Kontokostas, "Shapes Constraint Language (SHACL)," *W3C*, 2017. [Online]. Available: https://www.w3.org/TR/shacl/.