1.1 SUMMARY

1.2 LITERATURE REVIEW

1.2.1 PROVENANCE FOR THE SEMANTIC WEB

In his book *Weaving the Web: the original design and ultimate destiny of the World Wide Web*, Tim Berners Lee, the inventor of the WWW, states:

> *I have a dream for the Web [in which computers] become capable of analysing all the data on the Web – the content, links, and transactions between people and computers. A "Semantic Web", which makes this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The "intelligent agents" people have touted for ages will finally materialise*. (Berners-Lee, Weaving the Web: the original design and ultimate destiny of the World Wide Web, 1999)

In this vision, which represents the first formulation of the Semantic Web, it is already possible to identify the criticality that in the following years would have led to discuss the provenance topic. The data must be reliable in a world where automatic data analysis systems manage trade, bureaucracy, and daily lives. However, the Web is an open and inclusive dimension in which it is possible to find contradictory and questionable information. Therefore, it is essential to own indications such as the primary source of data, who created or modified it, and when that happened. However, the underlying technologies of the Semantic Web (RDF, OWL, SPARQL) were not originally intended to express such information.

To revise state of the art and develop a roadmap on provenance for Semantic Web technologies, the Provenance Incubator Group (Provenance Incubator Group Charter, 2010) was established in 2010. One of the first problems was identifying a shared and universal definition of "provenance", a task that proved impossible given its broad and multisectoral nature. Therefore, a working definition was accepted, restricted the context of the Web:

> *Provenance of a resource is a record that describes entities and processes involved in producing and delivering or otherwise influencing that resource. Provenance provides a critical foundation for assessing authenticity, enabling trust, and allowing reproducibility. Provenance assertions are a form of contextual metadata and can themselves become important records with their own provenance*. (Gil, et al., 08 December 2010)

Starting from this working definition, the group has compiled 33 use cases to formulate scenarios and requirements. Topics covered included eScience, eGovernment, business, manufacturing, cultural heritage, and library science, to name a few. The analysis of these use cases led to the elaboration of three scenarios: a news aggregator, the study of an epidemic and a business contract. The second scenario, the study of the epidemic, is particularly interesting for the case study of this work because it focuses on the reuse of scientific data. Alice is an epidemiologist studying the spread of a new disease called owl flu. Alice needs to integrate structured and unstructured data from different sources, to understand how data has evolved through provenance and version information. In addition, she needs to justify the results obtained by supporting the validity of the sources used, reusing data published by others in a new context and using the provenance to repeat previous analyses with new data. Introducing the problem with a concrete and complex example is helpful to understand how multifaceted and multidimensional it is. Specifically, provenance can be evaluated under three categories: content, management, and usage, each with various dimensions summarised in Table 1.

CATEGORY	DIMENSION	DESCRIPTION
CONTENT	Object	The artefact that a provenance statement is about.
	Attribution	The sources or entities that contributed to creating the artefact in question.
	Process	The activities (or steps) that were carried out to generate or access the artefact at hand.
	Versioning	Records of changes to an artefact over time and what entities and processes were associated with those changes.
	Justification	Documentation recording why and how a particular decision is made.
	Entailment	Explanations showing how facts were derived from other facts.
MANAGEMENT	Publication	Making provenance available on the Web.
	Access	The ability to find the provenance for a particular artefact.
	Dissemination	Defining how provenance should be distributed and its access be controlled.
	Scale	Dealing with large amounts of provenance.
USE	Understanding	How to enable the end-user consumption of provenance.
	Interoperability	Combining provenance produced by multiple different systems.
	Comparison	Comparing artefacts through their provenance.
	Accountability	Using provenance to assign credit or blame.
	Trust	Using provenance to make trust judgments.
	Imperfections	Dealing with imperfections in provenance records.
	Debugging	Using provenance to detect bugs or failures of processes.

Table 1 Dimensions of provenance

Historically, many data models, annotation frameworks, vocabularies and ontologies have been introduced to meet the above requirements. A complete list of all existing strategies will be drawn up in section 1.2.2, and their advantages and disadvantages will be explained.

1.2.2 REPRESENTING PROVENANCE IN RDF

The landscape of strategies to formally represent provenance in RDF data is vast and fragmented (Table 2). There are many approaches, varying in semantics, tuple typology, standard compliance, dependence on external vocabulary, blank node management, granularity and scalability. For an in-depth study of this topic, consult the article *Provenance-Aware Knowledge Representation: A Survey of Data Models and Contextualized Knowledge Graphs* (Sikos & Philp, 2020). First, the annotation syntaxes and, subsequently, the knowledge organisation systems related to provenance will be discussed in sections 1.2.2.1 and 1.2.2.2.

TYPE OF APPROACH	METADATA REPRESENTATION MODELS
Quadruples	Named graphs, RDF/S graphsets, RDF triple coloring
Extension of the RDF data model	Notation 3 Logic, RDF$^+$, annotated RDF (aRDF) and Annotated RDF Schema, SPOTL(X), RDF*
Encapsulating Provenance with RDF Triples	PaCE, singleton property
Data models alternative to RDF	GSMM, mapping entities to vectors
Knowledge organisation system	OPM, PML, Provenir, PREMIS, SWAN, DC, PROV, OCDM

Table 2 Annotation frameworks for RDF provenance

1.2.2.1 ANNOTATION SYNTAXES FOR RDF PROVENANCE

To date, the only standard syntax for annotating triples' provenance is *RDF reification* and is the only one to be compatible with all RDF-based systems. Included since RDF 1.0 (Manola & Miller, 2004), it consists in associating a statement to a new node of type *rdf:Statement*, which is connected to the triple by the predicates *rdf:subject*, *rdf:predicate*, and *rdf:object*. For example, consider the statement (*exproducts:item10245*, *exterms:weight*, *"2.4" xsd:decimal*) (Figure 1). Using the reification vocabulary, a new node *exproducts:triple12345* is introduced and associated with the following properties:

- (*exproducts:triple12345, rdf:type*, *rdf:Statement*)
- (*exproducts:triple12345, rdf:subject*, *exproducts:item10245*)
- (*exproducts:triple12345*, *rdf:predicate*, *exterms:weight*)
- (*exproducts:*item10245, rdf:*object*, *"2.4" xsd:decimal*).

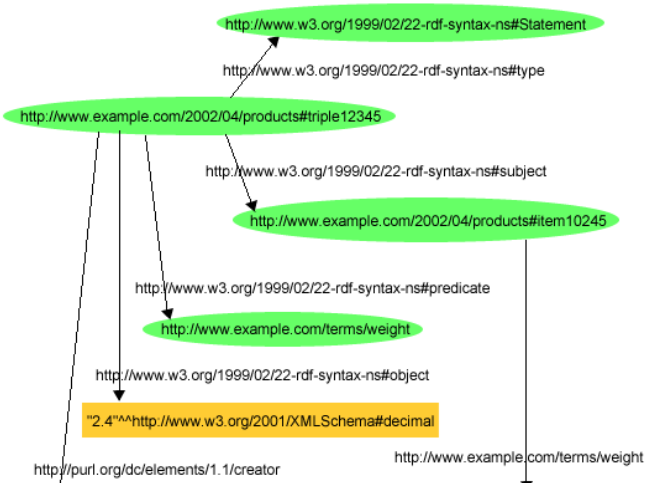

Figure 1 A statement, its reification, and its attribution

Finally, this new URI can become the subject of new provenance triples, as the responsible agent, expressed through the property (*exproducts:item10245*, *dc:creator*, *exstaff:85740*).

Such methodology has a considerable disadvantage: the size of the dataset is at least quadrupled since subject, predicate, and object must be repeated to add at least one provenance's information. There is a shorthand notation, the rdf:ID attribute in RDF/XML (Figure 2), but *RDF Reification* has no syntactic support for other serialisations.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:exterms="http://www.example.com/terms/"
    xml:base="http://www.example.com/2002/04/products">
    <rdf:Description rdf:ID="item10245">
        <exterms:weight rdf:ID="triple12345" rdf:datatype="&xsd;decimal">
            2.4
        </exterms:weight>
    </rdf:Description>
    <rdf:Description rdf:about="#triple12345">
        <dc:creator rdf:resource="http://www.example.com/staffid/85740"/>
    </rdf:Description>
</rdf:RDF>
```

Figure 2 Generating reifications using rdf:ID

Finally, composing SPARQL queries to obtain provenance annotated through *RDF Reification* is cumbersome: to identify the URI of the reification, it is necessary to explicit the entire reference triple. For all the mentioned reasons, there are several deprecation proposals for this syntax, including that by David Beckett, one of the editors of RDF in 2004 and RDF/XML (Revised) W3C Recommendation:

> *There are a few RDF model parts that should be deprecated (or removed if that seems possible), in particular reification which turned out not to be widely used, understood or implemented even in the RDF 2004 update* (Beckett, 2010).

After *RDF Reification*, in 2006, the W3C published a note that suggested a new approach to express provenance, called *n-ary relations* (Noy & Rector, 2006). In RDF and OWL, properties are always binary relationships between two URIs or a URI and a value. However, sometimes it is convenient to connect a URI to more than one other URI or value, such as expressing the provenance of a certain relationship. The *n-ary relations* allow this behaviour through the instance of a relationship in the form of a blank node. Taking the above example, (*exproducts:item10245*, *exterms:weight*, *"2.4" xsd:decimal*) becomes (*exproducts:item10245*, *exterms:weight*, *_:Weight*). Then, *_:Weight* can be associated with the value by (*_:Weight*, *exterms:weight*, *"2.4"xsd:decimal*), and to the provenance by (*_:Weight*, *dc:creator*, *exstaff:85740*). From an ontological point of view, the *_:Weight* class is a "reified relationship". Therefore, there is a clear similarity between *n-ary relations* and *RDF Reification*, with the difference that the latter reifies the statement, the first the predicate, with the advantage of not having to repeat all the triple elements but only the predicate. The second similarity, which is the main disadvantage of *n-ary relations*, is the need to introduce blank nodes, which cannot be globally dereferenced.

In summary, *RDF Reification* and *n-ary relations* are the only standard and the only alternative recommended by W3C to describe the provenance in RDF and have fatal design flaws. For this reason, different approaches have been

proposed since 2005, starting with Named Graphs and *formulae* in Notation 3 Logic, as will be clarified in the following paragraphs.

Named Graphs are graphs associated with a name in the form of a URI. They allow RDF statements describing graphs, with multiple advantages in numerous applications. For example, in Semantic Web publishing, named graphs allow a publisher to sign its graphs so that different information consumers can select specific graphs based on task-specific trust policies. Different tasks require different levels of trust. A naive information consumer may, for example, decide to accept any graph, thus collecting more information as well as false information. Instead, a more cautious consumer may require only graphs signed by known publishers, collecting less but more accurate data (Carroll, Bizer, Hayes, & Stickler, 2005). From a syntactical point of view, named graphs are quadruples, where the fourth element is the graph URI that acts as context to triples. It is a solution compatible with the RDF data model, does not rely on terms or ontologies to capture the provenance, does not cause *triple bloat*, and is therefore scalable and suitable for Big Data applications. On the other hand, concerning serialisation, it is possible to implement named graphs using extensions of RDF/XML, Turtle and N-Triples, called respectively TriX, TriG and N-Quads, all standardised and compatible with the SPARQL algebra.

The above advantages have led the Web Alliance to propose named graphs as a format to express the provenance of scientific statements. The suggested model is called “nanopublications” and represents a fundamental scientific statement with associated context. Precisely, a nanopublication consists of three named graphs: one on data, one on provenance and one on publication metadata (Groth, Gibson, & Velterop, 2010).

However, named graphs have a limit: they do not allow managing the provenance of implicit triples in the presence of update queries. RDFS allows the addition of semantics to RDF triples, so it is possible, through inference rules, to derive new implicit triples that are not explicitly declared. The moment an update query erases a named graph, all the logic of the triple associates get lost along with the data, and there is no way to separate the two aspects. *RDF/S graphsets*, and its evolution *RDF triple coloring*, extend named graphs to allow RDFS semantics. A graphset is a set of named graphs. It is associated with a URI, preserving provenance information lost following an update, and registering co-ownership of multiple named graphs (Pediaditis, Flouris, Fundulaki, & Christophides, 2009). Similarly, *RDF triple coloring* allows managing scenarios in which the same data has different resources, but co-ownership is implicit (Flouris, Fundulaki, Pediaditis, Theoharis, & Christophides, 2009). To better understand the problem, Table 2 shows four quadruples whose fourth element is the "color", the source of the triple.

S	**P**	**O**	**“COLOR”**
TheWashingtonPost	rdf:type	Newspaper	C_4
Newspaper	rdf:type	rdfs:Class	C_3
Newspaper	rdfs:subClassOf	MassMedia	C_3
MassMedia	rdfs:subClassOf	Media	C_5

Table 3 RDF triple coloring example

From the statements in Table 2, it is possible to infer that *Newspaper* is a subclass of *Media* since *Newspaper* is a subclass of *Massmedia* and *Massmedia* is a subclass of *Media*. Thus the origin of the implicit statement *(Newspaper*, *rdfs:subClassOf*, *Massmedia)* is C_3 and C_5. Named graphs could express this double provenance only with two separate quadruples. However, the query "returns the triple coloured C_3" would falsely return *(Newspaper*, *rdfs:subClassOf*, *Massmedia)*, ignoring that the provenance is not C_3 but C_3 and C_5. *RDF triple coloring* solves the problem by introducing the operator +, such that $C_{3,5} = C_3 + C_5$. $C_{3,5}$ is a new URI assigned to those triples that have as their source both C_3 and C_5.

Both *RDF/S graphsets* and *RDF triple coloring* are serialisable in TriG, TriX and N-Quads, do not need proprietary terms or external vocabularies and are scalable. However, *RDF/S graphsets* does not comply with either the RDF data model or the SPARQL algebra, unlike *RDF triple coloring*, which is fully compatible.

On the other hand, quadruples are not the only strategy to correlate RDF triples with provenance information. Additionally, the RDF data model can be extended to achieve this goal. The first proposal of this kind was Notation 3 Logic, which introduced the *formulae* (Berners-Lee, Notation 3 Logic, 2005). *Formulae* allow producing statements on N3 sentences, which are encapsulated by the syntax {...}. Berners-Lee and Connolly also proposed a *patch file format* for RDF deltas, or three new terms, using N3 (Berners-Lee & Connolly, Delta: an ontology for the distribution of differences between RDF graphs, 2004):

1. diff:replacement, that allows expressing any change. Deletions can be written as {…} diff:replacement {}, and additions as {} diff:replacement {…}.
2. diff:deletion, which is a shortcut to express deletions as {…} diff:deletion {…}.
3. diff:insertion, which is a shortcut to express additions as {…} diff:insertion {…}.

The main advantage of this representation is its economy: given two graphs G1 and G2, its cost in storage is directly proportional to the difference between the two graphs. Therefore, it is a scalable approach. However, while conforming to the SPARQL algebra, N3 does not comply with the RDF data model and relies on the N3 Logic Vocabulary.

Adopting a completely different perspective, RDF$^+$ solves the problem by attaching a provenance property and its value to each triple, forming a quintuple (Table 4). In addition, it extends SPARQL with the expression "WITH META *Metalist*", which includes graphs specified in *Metalist*, containing RDF$^+$ meta knowledge statements (Dividino, Sizov, Staab, & Schueler, 2009). To date, RDF$^+$ is not compliant with any standard, neither the RDF data model, nor SPARQL, nor any serialisation formats.

SUBJECT	PREDICATE	OBJECT	META-PROPERTY	META-VALUE
:ra/15519	foaf:name	“Silvio Peroni”	:accordingTo	orcid:0000-0002-8420-0696

Table 4 An RDF$^+$ quintuple

Also, *SPOTL(X)* allows expressing a triple provenance through quintuple (Hoffart, Suchanek, Berberich, & Weikum, 2013). Indeed, the framework’s name means Subject Predicate Object Time Location. Optionally, it is possible to create sextuples that add context to the previous elements. *SPOTL(X)* is concretely implemented in YAGO, a knowledge base automatically built from Wikipedia, given the need to specify in which time, space and context a specific statement is true. Outside of YAGO, *SPOTL(X)* does not follow either the RDF data model or the SPARQL algebra, and there is no standard serialisation format.

Similarly, *annotated RDF* (aRDF) does not currently have any standardisation. A triple annotation has the form (s, p: λ, o), where λ is the annotation, always linked to the property (Udrea, Recupero, & Subrahmanian, 2010). *Annotated RDF Schema* perfects this pattern by annotating an entire triple and presenting a SPARQL extension to query annotations, called AnQL (Zimmermann, Lopes, Polleres, & Straccia, 2012). In addition, it specifies three application domains: the temporal, fuzzy and provenance domains (Table 5).

DOMAIN	ANNOTATED TRIPLE	MEANING
Temporal	(niklasZennstrom, ceoOf, skype): [2003, 2007]	Niklas was CEO of Skype during the period 2003 to 2007
Provenance	(niklasZennstrom, ceoOf, skype): wikipedia	Niklas was CEO of Skype according to Wikipedia
Fuzzy	(skype, ownedBy, bigCompany): 0.3	Skype is owned by a big company to a degree not less than 0.3
Temporal, provenance and fuzzy	(niklasZennstrom, ceoOf, skype): <[2003, 2007], 1, wikipedia>	Niklas was without doubt CEO of Skype during the period 2003 to 2007, according to Wikipedia

Table 5 Annotated RDF Schema application examples

The most recent proposal in extending the RDF data model to handle provenance information was RDF*, which embeds triples into triples as the subject or object (Hartig & Thompson, 2019). Its main goal is to replace *RDF Reification* through less verbose and redundant semantics. Since there is no serialisation to represent such syntax, Turtle*, an extension of Turtle to include triples in other triples within << and >>, has also been introduced. Similarly, SPARQL* is an RDF*-aware extension for SPARQL. Later, RDF* was proposed to allow statement-level annotations in RDF streams by extending RSP-QL to RSP-QL* (Keskisärkkä, Blomqvist, Lind, & Hartig, 2019).

After discussing possible RDF extension, two strategies encapsulate provenance in RDF triples: PaCE and singleton properties. Provenance Context Entity (Pace) is an approach concretely implemented in the Biomedical Knowledge Repository (BKR) project at the US National Library of Medicine (Sahoo, Bodenreider, Hitzler, Sheth, & Thirunarayan, 2010). Its implementation is flexible and varies depending on the application. It allows three granularity levels: the provenance can be linked to the subject, predicate and object of each triple, only to the subject or only to the subject and predicate, through the property *provenir:derives_from*. Therefore, such a solution depends on the Provenir ontology, and it is not scalable because it causes *triple bloat*. Apart from these two flaws, it has several advantages: it leads to 49% less triple than *RDF Reification*, does not involve blank nodes, is fully compatible with the RDF data model and SPARQL, and allows serialisation in any RDF format (RDF/XML, N3, Turtle, N-Triples, RDF-JSON, JSON-LD, RDFa and HTML5 Microdata).

Conversely, singleton properties are inspired by set theory, where a singleton set has a single element. Similarly, a singleton property is defined as “a unique property instance representing a newly established relationship between two existing entities in one particular context” (Nguyen, Bodenreider, & Sheth, 2014). This goal is achieved by connecting subjects to objects with unique properties that are singleton properties of the generic predicate via the new *singletonPropertyOf* predicate. Then, meta-knowledge can be attached to the singleton property (Table 6). This strategy has been shown to have advantages in terms of query size and query execution time over PaCE (tested on BKR) but disadvantages in terms of triples’ number where multiple predications share the same source. Beyond that, singleton properties have the same advantages and disadvantages as PaCE: they rely on a non-standard term, are not scalable, adhere to the RDF data model and SPARQL, and are serialisable in any RDF format.

SUBJECT	PREDICATE	OBJECT
:ra/15519	:name#1	“Silvio Peroni”
:name#1	:singletonPropertyOf	foaf:name
:name#1	:accordingTo	orcid:0000-0002-8420-0696

Table 6 Singleton property and its meta knowledge assertion example

Table 7 summarises all the considerations made so far on the advantages and disadvantages of the listed RDF-based strategies.

APPROACH	TUPLE TYPE	COMPLIANCE WITH THE RDF DATA MODEL	COMPLIANCE WITH SPARQL	RDF SERIALISATIONS	EXTERNAL VOCABULARY	SCALABLE
Named graphs	Quadruple	+	+	TriG, TriX, N-Quads	-	+
RDF/S graphsets	Quadruple	-	-	TriG, TriX, N-Quads	-	+
RDF triple coloring	Quadruple	+	+	TriG, TriX, N-Quads	-	+
N3Logic	Triple (in N3)	-	+	N3	N3 Logic Vocabulary	+
aRDF	Nonstandard	-	-	-	-	+
Annotated RDF Schema	Nonstandard	-	-	-	-	+
RDF^{+}	Quintuple	-	-	-	-	-
SPOTL(X)	Quintuple/sextuple	-	-	-	-	-
RDF*	Nonstandard	-	-	-	-	-
PaCE	Triple	+	+	RDF/XML, N3, Turtle, N-Triples, RDF-JSON, JSON-LD, RDFa, HTML5 Microdata	Provenir ontology	-
Singleton property	Triple	+	+	RDF/XML, N3, Turtle, N-Triples, RDF-JSON, JSON-LD, RDFa, HTML5 Microdata	*singletonPropertyOf* property	-

Finally, there are data models alternative to RDF to organise knowledge. The General Semistructured Meta-model (GSMM) is a meta-model to aggregate heterogeneous data models into a single formalism to manage them in a homogeneous way or compare (Damiani, Oliboni, Quintarelli, & Tanca , 2019). A triple-based database can be converted to a GSMM graph by introducing nodes for subjects, predicates and objects, where predicates have an incoming edge labelled < TO >, and an incoming edge labelled < FROM >. Since predicates are modelled as nodes, GSMM supports reification and allows to represent provenance. On the other hand, research exists to convert knowledge bases' entities into embeddings in a vector space (Suchanek, Lajus, Boschin, & Weikum, 2019). With embeddings, impossible operations in RDF representations can be performed, such as predicting links (using neural networks) or new facts (using logical rules).

1.2.2.2 KNOWLEDGE ORGANISATION SYSTEMS FOR RDF PROVENANCE

Historically, many vocabularies and ontologies have been introduced to represent provenance information, either upper ontologies, domain ontologies, and provenance-related ontologies. Among the upper ontologies, the Open Provenance Model stands out because of its interoperability. It describes the history of an entity in terms of processes, artefacts and agents (Moreau, et al., 2011), a pattern that will be discussed later on the PROV Data Model (Moreau & Missier, 2013). On the other hand, the Proof Markup Language (PML) is an ontology designed to support trust mechanisms between heterogeneous web services (Pinheiro da Silva, McGuinness, & Fikes, 2006).

About domain-relevant models, there is the Provenir Ontology for eScience (Sahoo & Sheth, Provenir Ontology: Towards a Framework for eScience Provenance Management, 2009), PREMIS for archived digital objects, such as files, bitstreams and aggregations (Caplan, 2017) and Semantic Web Applications in Neuromedicine (SWAN) Ontology to model a scientific discourse in the context of biomedical research (Ciccarese, et al., 2008). Finally, the Dublin Core Metadata Terms allows to express the provenance of a resource and specify what is described (e.g. dct:BibliographicResource), who was involved (e.g. dct:Agent), when the changes occurred (e.g. dct:dateAccepted), and the derivation (e.g. dct:references), sometimes very precisely (DCMI Metadata Terms, 2020).

All the requirements and ontologies mentioned have been merged into a single data model, the PROV Data Model (Moreau, et al., 2011), translated into the PROV Ontology using the OWL 2 Web Ontology Language (Lebo, Sahoo, & McGuinness, 2013). It provides several classes, properties, and restrictions, representing provenance information in different systems and contexts. Its level of genericity is such that it is even possible to create new classes and data model-compatible properties for new applications and domains. Just like the Open Provenance Model, PROV-DM captures the provenance under three complementary perspectives:

- *Agent-centred provenance*, which people, organisations, software, inanimate objects, or other entities are involved in the generation, manipulation, or influence of a resource. For example, concerning a journal article, it is possible to distinguish between the author, the editor, and the publisher. PROV-O maps the responsible agent with prov:Agent, the relationship between an activity and the agent with prov:wasAssociatedWith and an entity's attribution to an agent with prov:wasAttributedTo.
- *Object-centred-provenance*, which is the origin of a document's portion from other documents. Taking the example of the article, a fragment of it can quote an external document. PROV-O maps a resource with prov:Entity, whether physical, digital, or conceptual, while the predicate prov:wasDerivedFrom expresses a derivation relationship.
- *Process-centred provenance*, or the actions and processes necessary to generate a resource. For example, an editor can edit an article to correct spelling errors using the previous version of the document. PROV-O expresses the concept of action with prov:Activity, creating an entity with the predicate prov:wasGeneratedBy and the use of another entity to complete a passage with prov:used.

The diagram in Figure 1 provides a high-level view of the discussed concepts' structure, constituting the so-called "starting point terms". PROV-O is much more extensive and provides sophisticated entities, agents, activities, and relationships in a modular way, namely "expanded terms" and "qualified terms".

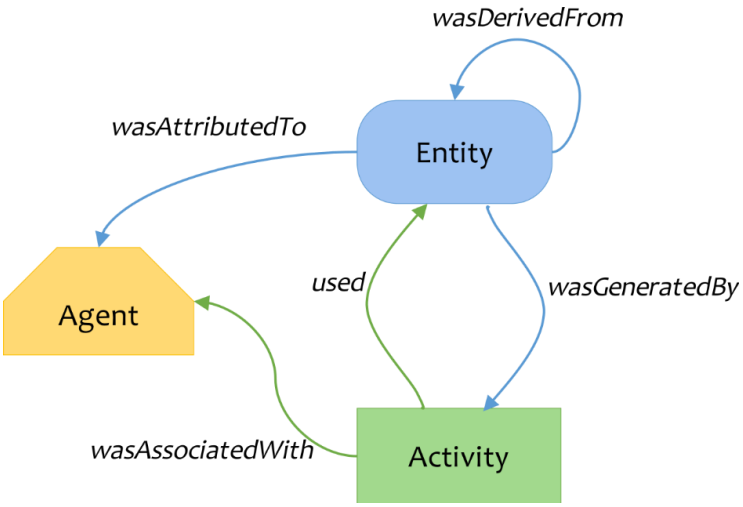

Figure 3 High level overview diagram of PROV records (Gil & Miles, PROV Model Primer, 2013)

The OpenCitations Data Model, used in this research, relies on the flexibility of PROV-O to record the provenance of bibliographic datasets (Daquino, et al., 2020). Each bibliographical entity described by the OCDM is annotated with one or more snapshots of provenance. The snapshots are of type prov:Entity and are connected to the bibliographic entity described through prov:specializationOf, predicate present in the mentioned "expanded terms". Being the specialisation of another entity means sharing every aspect of the latter and, in addition, presenting more specific aspects, such as an abstraction, a context or, in this case, a time. In addition, each snapshot records the validity dates (prov:generatedAtTime, prov:invalidatedAtTime), the agents responsible for both creation and modification of the metadata (prov:wasAttributedTo), the primary sources (prov:hadPrimarySource) and a link to the previous snapshot in time (prov:wasDerivedFrom). The model is summarised in Figure 2.

In addition, OCDM extends the Provenance Ontology by introducing a new property called *hasUpdateQuery*, a mechanism to record additions and deletions from an RDF graph with a SPARQL INSERT and SPARQL DELETE query string. The *snapshot-oriented* structure, combined with a system to explicitly indicate how a previous snapshot was

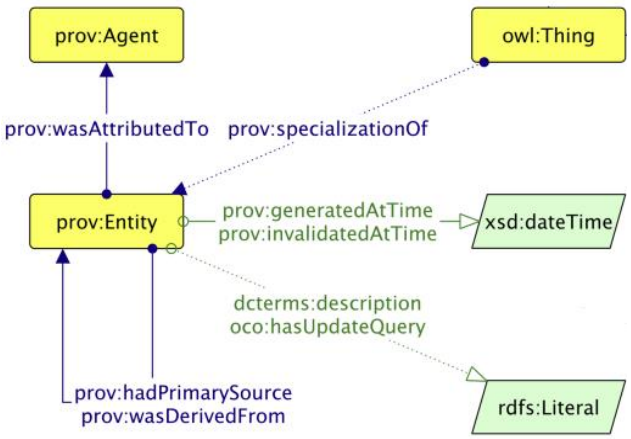

Figure 4 Provenance in the OpenCitations Data Model

modified to reach the current state, makes it easier to recover the current statements of an entity and restore an entity to a specific snapshot. The current statements are those available in the present dataset, while recovering a snapshot s_i means applying the reverse operations of all update queries from s_n to s_{i+1} (Peroni, Shotton, & Vitali, 2016).

This expedient, initially adopted for the OpenCitations Corpus, was designed to foster reusability in other contexts and is the added value of the provenance model proposed in the OCDM, which is the basis for the library to perform time agnostic queries presented in this work.

In the next section, the existing literature on tracking changes in RDF data will be deepened, focusing on the sources that inspired the OpenCitations provenance model.

1.2.3 TRACKING CHANGES OF RDF DATA

The different metadata representation models discussed in section 1.2.2 can be used in various ways and even combined according to the implementation requirements of a specific repository. From an application point of view, there are two main approaches to keeping track of how RDF data changes over time, according to the taxonomy proposed in the article *A document-inspired way for tracking changes of RDF data* (Peroni, Shotton, & Vitali, 2016). On the one hand, there are *statement-centric* strategies; on the other *resource-centric* methods. S*tatement-centric* strategies report changes to the entire dataset, that is, to the entire set of statements in a dataset, while *resource-centric* strategies report changes to a specific resource. Among the *statement-centric* approaches, there are two subcategories: *physical snapshot* and *massive statement reification*.

Physical snapshots refer to creating a copy of a LOD dataset's entire set of statements at arbitrary regular time intervals, such as every week or month. The only advantage of this model is its ease of implementation. The disadvantages are much more consistent: first, a massive amount of space and time is needed to store all copies of a dataset; furthermore, many changes may not be reflected in the snapshots, that is, all statements with a lifespan shorter than the interval between snapshots. Given the different statements' versions, further diff mechanisms are needed to identify what has changed.

1.2.4 COMPUTING THE DELTA BETWEEN RDF DATA

Recording document changes is a problem that goes beyond RDF and has its origins in the notion of the delta. In the article *Introduction to the Universal Delta Model*, Gioele Barabucci defines the delta as:

> *A delta $\Delta_{S,T}$ is a tuple of changes (C) and change relations (R) that describes how to transform the source document (S) into the target document (T)* (Barabucci, Introduction to the Universal Delta Model, 2013).

$$\Delta_{S,T} \equiv (C, R)$$

Since all documents are linear at the bitstream level, it is theoretically possible to apply the same delta algorithms for textual documents to RDF data. However, existing version control systems, such as RCS and CVS, are not a workable solution to compute RDF deltas for two reasons. First, the implementation cannot depend on the text format since RDF is an abstract model serialisable in several formats. Furthermore, the lower the compared abstraction level, the less meaningful the delta produced. The meaningfulness indicates how much delta concision is due to complex changes, that is, to a high level of abstraction (Barabucci, Ciancarini, Iorio, & Vitali, 2016). An RDF document is a *graphical document* in which relationships of any kind link the elements, which means that the information is modelled as a graph at a higher abstraction level.

Therefore, there are at least two problems to solve to get a significant delta for RDF documents, which will be explored one at a time:

1. It is necessary to define the level of abstraction to consider to compute the differences between two graphs.
2. Specific algorithms shall be introduced to obtain the established output given the input.

Considering a single RDF document, the highest level of abstraction and granularity is the document itself. However, using the document to compute a change leads to a coarse output, full of irrelevant information that unnecessarily subtracts RAM and storage space. This argument also applies to the immediately lower level of granularity, the named graph. On the other hand, using a triple as granularity level conducts to a correct result only in the absence of blank nodes, while it fails in case two triples share the same blank node. The reason is apparent by looking at the example in Figure 3, where two triples containing blank nodes represent two different people with the same surname, not the same person who changes name.

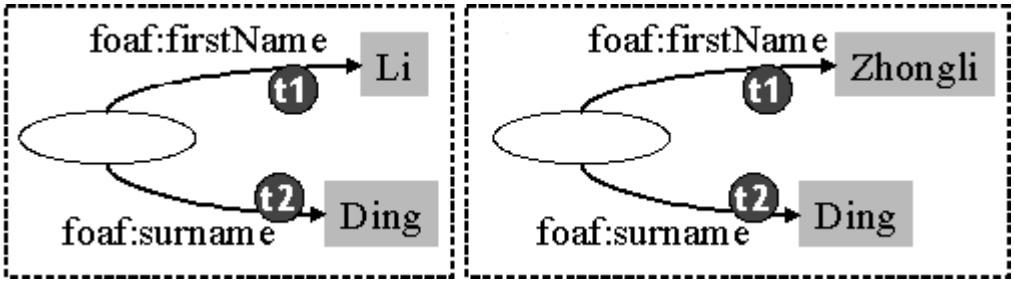

Figure 5 Two triples with blank nodes representing two different people.

Therefore, it is necessary to identify an intermediate level between the triple and the named-graph, a so-called "RDF molecule", that is "the finest and lossless component of an RDF graph" (Ding, Finin, Peng, Silva, & L., 2005). A sub-graph is *lossless* if it can be used to restore the original graph without introducing new triples, while it is said *finest* if it cannot be further decomposed into lossless sub-graphs. This principle was applied in the Opencitations Data Model by choosing the graph associated with an entity as a contextual space to compute the delta and record the provenance.

Once the level of granularity is identified, that is, the input of the diff algorithm, an output format must be chosen. A first formal model to represent changes in RDF repositories was proposed in 2002 and is based on four assumptions (Ognyanov & Kiryakov, 2002):

- *The RDF statement is the smallest directly manageable piece of knowledge*. It is not possible to add, remove or edit a resource without changing at least one statement.
- *An RDF statement cannot be changed — it can only be added and removed*. Only the constituents of a triple determine its identity and, if the constituents change, the triple is converted to a new triple.
- *The two basic types of updates in a repository are the addition and removal of a statement*. Therefore, only these two events need to be tracked by a tracking system. If more complex modifications occur, such as substitution or simultaneous additions, these must be treated and represented in their atomic operations.
- *Each update turns the repository into a new state*. Since a repository state is determined by its statements set, the repository is turned into a new state when an update changes the statements.

The fourth principle derives a corollary: the history of changes in a repository is the sequence of its states. If equivalent states exist with isomorphic graphs, they are treated as different states at different times. Moreover, some repository states can be marked as versions, depending on the user and the application's needs.

Based on the four assumptions and the corollary, the article by Ognyanov and Kiryakov finally proposes an implementation approach. It involves an *update counter* (UC), which increases its integer variable each time the repository is updated, to form an *update identifier* (UID). For each state, the added or deleted statements are indicated, according to the format: **UID:nn {add|remove} <subj, pred, obj>** (Figure 4).

```
History:
UID:1 add <A, r1, B>
UID:2 add <E, r1, D>
UID:3 add <E, r3, B>
UID:4 add <D, r3, A>
UID:5 add <C, r2, D>
UID:6 add <A, r2, E>
UID:7 add <C, r2, E>
UID:8 remove <A, r2, E>
UID:9 add <B, r2, C>
UID:10 remove <E, r3, B>
UID:11 remove <B, r2, C>
UID:12 remove <C, r2, E>
UID:13 remove <C, r2, D>
UID:14 remove <E, r1, D>
UID:15 remove <A, r1, B>
UID:16 remove <D, r3, A>
```

Figure 6 Representation of the history of an RDF repository according to the model of Ognyanov e Kiryakov

The Opencitations Data Model incorporates many of the concepts contained in the Ognyanov and Kiryakov model: it records changes as deletions and additions of statements and, at each update, turns the provenance graph of the related entity to a new state, numbered by an integer, similar to the UID. Each provenance graph is identified by a URL in the form of [snapshot entity URL]:[entity provenance URL]se/[iterative number], e.g.: https://w3id.org/oc/corpurs/br/1423490/prov/se/2.

However, the Ognyanov and Kiryakov's model has a limit. In case of multiple additions and deletions, the repository is turned to a different state for every single statement added or removed. Berners-Lee and Connolly solved this problem, proposing in 2004 a patch file format for RDF deltas, or three new terms (Berners-Lee & Connolly, Delta: an ontology for the distribution of differences between RDF graphs, 2004):

4. diff:replacement, that allows expressing any change. Deletions can be written as {…} diff:replacement {}, and additions as {} diff:replacement {…}.
5. diff:deletion, which is a shortcut to express deletions as {…} diff:deletion {…}.

6. diff:insertion, which is a shortcut to express additions as {…} diff:insertion {…}.

The main advantage of this representation is its economy: given two graphs G1 and G2, its cost in storage is directly proportional to the difference between the two graphs.

The OpenCitations Data Model takes up Berners-Lee and Connolly's proposal, perfecting it using SPARQL 1.1 to express additions and deletions uniquely. It is important to note that the first version of SPARQL was released in 2008, four years after Berners-Lee and Connolly's article, while SPARQL 1.1, which introduced the INSERT and DELETE operations, is 2013.

Once the granularity level has been established, an algorithm to compute the difference between two RDF graphs must be defined. In this respect, it is necessary to distinguish between graphs with or without blank nodes. Given two graphs G1 and G2, if each triple's node within them is either a literal or a URI, the two graphs are *grounded* and computing their difference is immediate and straightforward.

> *If G1 and G2 are ground RDF graphs, then the ground graph delta of G1 and G2 is a pair (insertions, deletions) where insertions is the set difference G2 − G1 and deletions is G1 − G2* (Berners-Lee & Connolly, Delta: an ontology for the distribution of differences between RDF graphs, 2004).

If blank nodes are present, the discourse is more complex: in the absence of an ontology, it is impossible to compute the diff without the risk of generating inconsistencies. Then, it is necessary to derive a so-called *fully labelled* graph, in which every node is *functionally ground* to an ontology. For a node *n* to be functionally grounded, at least one of the following conditions must occur:

- There is a triple (n, p, o) in G, p is an inverse functional property according to an ontology W, and either *o* is grounded or functionally grounded. In other words, it is possible to disambiguate *n* since, given *o* and W, there is only one possible value of *n*.
- There is a triple (*s*, p, n) in G, p is a functional property according to an ontology W, and *s* is grounded or functionally grounded. In other words, it is possible to disambiguate *n* since, since given *s* and W, there is only one possible value of *n*.
- There is a node *n'* in G such that *n* is equivalent to *n'* compared to W, and *n* is grounded or functionally grounded.

In the presence of fully labelled graphs, context-free "strong" diffs can be obtained, similarly as seen for ground RDF deltas:

> *Given a background ontology W, a strong delta between fully labelled graphs G1 and G2 is a pair (insertions, deletions) where insertions is the set difference F2 − F1, deletions is F1 − F2, and F1 and F2 are functional analogs of G1 and G2 respectively* (Berners-Lee & Connolly, Delta: an ontology for the distribution of differences between RDF graphs, 2004).

1.3 BIBLIOGRAPHY

Barabucci, G. (2013). Introduction to the Universal Delta Model. *Proceedings of the 2013 ACM Symposium on Document Engineering* (p. 47–56). Florence, Italy: Association for Computing Machinery. doi:10.1145/2494266.2494284

Barabucci, G., Ciancarini, P., Iorio, A. D., & Vitali, F. (2016). Measuring the quality of diff algorithms: a formalisation. *Computer Standards & Interfaces, 46*, 52-65. doi:10.1016/j.csi.2015.12.005

Beckett, D. (2010, April 10). *RDF Syntaxes 2.0.* Retrieved 07 22, 2021, from W3C: https://www.w3.org/2009/12/rdf-ws/papers/ws11

Berners-Lee, T. (1999). *Weaving the Web: the original design and ultimate destiny of the World Wide Web.* San Francisco: Harper San Francisco.

Berners-Lee, T. (2005, August). *Notation 3 Logic.* Retrieved 07 23, 2021, from W3C: https://www.w3.org/DesignIssues/N3Logic

Berners-Lee, T., & Connolly, D. (2004). Delta: an ontology for the distribution of differences between RDF graphs. Retrieved from https://www.w3.org/DesignIssues/lncs04/Diff.pdf

Caplan, P. (2017). Understanding PREMIS: an overview of the PREMIS Data Dictionary for Preservation Metadata. Library of Congress. Retrieved from Library of Congress: https://www.loc.gov/standards/premis/understanding-premis-rev2017.pdf

Carroll, J. J., Bizer, C., Hayes, P., & Stickler, P. (2005). Named graphs, provenance and trust. *Proceedings of the 14th international conference on World Wide Web* (p. 613–622). New York: Association for Computing Machinery. doi:10.1145/1060745.1060835

Ciccarese, P., Wu, E., Kinoshita, J., Wong, G. T., Ocana, M., Ruttenberg, A., & Clark, T. (2008). The SWAN Scientific Discourse Ontology. *Journal of biomedical informatics, 41*(5), 739–751. doi:10.1016/j.jbi.2008.04.010

Damiani, E., Oliboni, B., Quintarelli, E., & Tanca , L. (2019). A graph-based meta-model for heterogeneous data management. *Knowledge and Information Systems, 61*(1), 107–136. doi:10.1007/s10115-018-1305-8

Daquino, M., Peroni, S., Shotton, D., Colavizza, G., Ghavimi, B., Lauscher, A., . . . Zumstein, P. (2020). The OpenCitations Data Model. In J. Z. Pan, V. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, . . . L. Kagal (A cura di), *International Semantic Web Conference. 12507*, p. 447-463. Springer, Cham. doi:10.1007/978-3-030-62466-8_28

DCMI Metadata Terms. (2020, 01 20). Retrieved 07 16, 2021, from Dublin Core Metadata Initiative: http://dublincore.org/specifications/dublin-core/dcmi-terms/2020-01-20/

Ding, L., Finin, T., Peng, Y., Silva, P. P., & L., D. (2005). *Tracking RDF Graph Provenance.* Technical report. Retrieved from http://ebiquity.umbc.edu/get/a/publication/178.pdf

Dividino, R., Sizov, S., Staab, S., & Schueler, B. (2009). Querying for provenance, trust, uncertainty and other meta knowledge in RDF. *Journal of Web Semantics, 7*(3), 204-219. doi:10.1016/j.websem.2009.07.004

Flouris, G., Fundulaki, I., Pediaditis, P., Theoharis, Y., & Christophides, V. (2009). Coloring RDF Triples to Capture Provenance. *The Semantic Web - ISWC 2009.* Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-04930-9_13

Gil, Y., & Miles, S. (Eds.). (2013, 04 30). *PROV Model Primer.* Retrieved 07 16, 2021, from W3C: http://www.w3.org/TR/2013/NOTE-prov-primer-20130430/

Gil, Y., Cheney, J., Groth, P., Hartig, O., Miles, S., Moreau, L., & Pinheiro da Silva, P. (08 December 2010). *Provenance XG Final Report.* W3C. Retrieved from http://www.w3.org/2005/Incubator/prov/XGR-prov-20101214/

Groth, P., Gibson, A., & Velterop, J. (2010, 09 21). The anatomy of a nanopublication. *Information Services & Use, 30*(1-2), 51-56. doi:10.3233/ISU-2010-0613

Hartig, O., & Thompson, B. (2019, March 20). Foundations of an Alternative Approach to Reification in RDF. Retrieved from https://arxiv.org/abs/1406.3399

Hoffart, J., Suchanek, F. M., Berberich, K., & Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base. *Artificial Intelligence, 194*, 28-61. doi:10.1016/j.artint.2012.06.001

Keskisärkkä, R., Blomqvist, E., Lind, L., & Hartig, O. (2019, November 04). RSP-QL* : Enabling Statement-Level Annotations in RDF Streams. *The Power of AI and Knowledge Graphs. SEMANTiCS 2019. Lecture Notes in Computer Science. 11702.* Karlsruhe, Germany: Springer, Cham. doi:10.1007/978-3-030-33220-4_11

Lebo, T., Sahoo, S., & McGuinness, D. (Eds.). (2013, 04 30). *PROV-O: The PROV Ontology.* Retrieved 07 16, 2021, from W3C: http://www.w3.org/TR/2013/REC-prov-o-20130430/

Manola, F., & Miller, E. (Eds.). (2004, February 10). *RDF Primer.* Retrieved 07 22, 2021, from W3C: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

Moreau, L., & Missier, P. (Eds.). (2013, 04 30). *PROV-DM: The PROV Data Model.* Retrieved 07 16, 2021, from W3C: http://www.w3.org/TR/2013/REC-prov-dm-20130430/

Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., . . . Bussche, J. V. (2011). The Open Provenance Model core specification (v1.1). *Future Generation Computer Systems,, 27*(6), 743-756. doi:10.1016/j.future.2010.07.005

Nguyen, V., Bodenreider, O., & Sheth, A. (2014). Don't like RDF reification?: making statements about statements using singleton property. *WWW '14: Proceedings of the 23rd international conference on World wide web* (pp. 759–770). New York, NY, USA: Association for Computing Machinery. doi:10.1145/2566486.2567973

Noy, N., & Rector, A. (Eds.). (2006, 04 12). *Defining N-ary Relations on the Semantic Web.* Retrieved 07 22, 2021, from W3C: http://www.w3.org/TR/2006/NOTE-swbp-n-aryRelations-20060412/

Ognyanov, D., & Kiryakov, A. (2002). Tracking Changes in RDF(S) Repositories. In G.-P. A., & B. V.R. (Ed.), *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web* (pp. 373-378). Berlin, Heidelberg: Springer. doi:10.1007/3-540-45810-7_33

Pediaditis, P., Flouris, G., Fundulaki, I., & Christophides, V. (2009). On Explicit Provenance Management in RDF/S Graphs. *First Workshop on the Theory and Practice of Provenance.* San Francisco, CA, USA: USENIX. Retrieved from https://www.usenix.org/legacy/event/tapp09/tech/full_papers/pediaditis/pediaditis.pdf

Peroni, S., Shotton, D., & Vitali, F. (2016). A Document-inspired Way for Tracking Changes of RDF Data. In L. Hollink, S. Darányi, A. M. Peñuela, & E. Kontopoulos (Ed.), *Detection, Representation and Management of Concept Drift in Linked Open Data. 1799*, pp. 26-33. Bologna: CEUR Workshop Proceedings. Retrieved from http://ceur-ws.org/Vol-1799/Drift-a-LOD2016_paper_4.pdf

Pinheiro da Silva, P., McGuinness, D. L., & Fikes, R. (2006). A proof markup language for Semantic Web services. *Information Systems, 31*(4–5), 381-395. doi:10.1016/j.is.2005.02.003

Provenance Incubator Group Charter. (2010). Retrieved July 15, 2021, from https://www.w3.org/2005/Incubator/prov/charter

Sahoo, S. S., & Sheth, A. P. (2009). Provenir Ontology: Towards a Framework for eScience Provenance Management. Retrieved from https://corescholar.libraries.wright.edu/knoesis/80

Sahoo, S. S., Bodenreider, O., Hitzler, P., Sheth, A., & Thirunarayan, K. (2010). Provenance Context Entity (PaCE): Scalable Provenance Tracking for Scientific RDF Data. In G. M., & L. B., *Scientific and Statistical Database Management* (Vol. 6187, pp. 461-470). Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-13818-8_32

Sikos, L., & Philp, D. (2020). Provenance-Aware Knowledge Representation: A Survey of Data Models and Contextualised Knowledge Graphs. *Data Science and Engineering, 5*(3), 293-316. doi:10.1007/s41019-020-00118-0

Suchanek, F. M., Lajus, J., Boschin, A., & Weikum, G. (2019). Knowledge Representation and Rule. In M. Krötzsch, & D. Stepanova (Ed.), *Reasoning Web. Explainable Artificial Intelligence: 15th International Summer School 2019, Bolzano, Italy, September 20-24, 2019, Tutorial Lectures* (pp. 110-152). Springer International Publishing. doi:10.1007/978-3-030-31423-1_4

Udrea, O., Recupero, D. R., & Subrahmanian, V. S. (2010, January). Annotated RDF. *ACM Transactions on Computational Logic, 11*(2), 1–41. doi:10.1145/1656242.1656245

Zimmermann, A., Lopes, N., Polleres, A., & Straccia, U. (2012). A general framework for representing, reasoning and querying with annotated Semantic Web data. *Journal of Web Semantics, 11*, 72-95. doi:10.1016/j.websem.2011.08.006