Modelling Provenance of DBpedia Resources Using Wikipedia Contributions

Fabrizio Orlandi a,*, Alexandre Passant a

a Digital Enterprise Research Institute, National University of Ireland, Galway, Ireland

Abstract

DBpedia is one of the largest datasets in the Linked Open Data cloud. Its centrality and its cross-domain nature makes it one of the most important and most referred to knowledge bases on the Web of Data, generally used as a reference for data interlinking. Yet, in spite of its authoritative aspect, there is no work so far tackling the provenance aspect of DBpedia statements. By being extracted from Wikipedia, an open and collaborative encyclopedia, delivering provenance information about it would help to ensure trustworthiness of its data, a major need for people using DBpedia data for building applications. To overcome this problem, we propose an approach for modelling and managing provenance on DBpedia using Wikipedia edits, and making this information available on the Web of Data.

In this paper, we describe the framework that we implemented to do so, consisting in (1) a lightweight modelling solution to semantically represent provenance of both DBpedia resources and Wikipedia content, along with mappings to popular ontologies such as the W7 — what, when, where, how, who, which, and why — and OPM — Open Provenance Model — models, (2) an information extraction process and a provenance-computation system combining Wikipedia articles’ history with DBpedia information, (3) a set of scripts to make provenance information about DBpedia statements directly available when browsing this source, as well as being publicly exposed in RDF for letting software agents consume it.

Key words: Provenance, Linked Data, DBpedia, Wikipedia

1. Introduction

Collaborative websites such as Wikipedia have recently shown the benefit of being able to create and manage very large public knowledge bases. However, one of the most common concerns about these types of information sources is the trustworthiness of their content which can be arbitrarily edited by everyone. The DBpedia project, which aims at converting Wikipedia content into structured knowledge, is then not exempt from this concern. Especially considering that one of the main objectives of DBpedia is to build a dataset such that Semantic Web technologies can be employed against it. Hence this allows not only to formulate sophisticated queries against Wikipedia, but also to link it to other datasets on the Web, or create new applications or mashups. Thanks to its large dataset (around 1 billion RDF triples) and its cross-domain

---

* Corresponding author. Tel: +353 91 494 035
Email addresses: fabrizio.orlandi@deri.org (Fabrizio Orlandi), alexandre.passant@deri.org (Alexandre Passant).
1 This work is funded by the Science Foundation Ireland under grant number SFI/08/CE/I1380 (Lion 2) and by an IRCSET scholarship.
2 http://www.wikipedia.org
3 Statistics about Wikipedia: http://stats.wikimedia.org/EN/Sitemap.htm
4 http://dbpedia.org/
nature DBpedia has become one of the most important and interlinked datasets on the Web of Data [4]. Therefore ensuring provenance information of DBpedia data is crucial, especially for developers consuming or interlinking its content.

Research on Wikipedia, and on collaborative websites in general, shows that some information quality aspects (such as currency and formality of language) of Wikipedia are quite high [5]. However, as suggested in [8], the high quality level of certain aspects of Wikipedia articles does not imply that it is good on other dimensions as well. In fact, a substantial qualitative difference exists in Wikipedia between “featured” articles (high quality articles identified by the community) and normal articles [8]. For this reason it is important to identify quality measures for Wikipedia articles and estimate the trustworthiness of their content. Then, since the DBpedia content is directly extracted from Wikipedia, the same trust and quality values can be propagated to the DBpedia dataset. However, in order to obtain these values, it is essential to provide detailed provenance information about the data published on the Web.

The benefits of using data provenance to develop trust on the Web, and the Semantic Web in particular, have been already widely described in the state of the art (see [6] and [7]). Provenance data provides useful information such as timeliness and authorship of data. It can be used as a ground basis for various applications and use cases such as identifying trust values for pages or pages fragments [2], or measuring users’ expertise by analysing their contributions [27] and then personalize trust metrics based on the user profile of a person on a particular topic [21]. Moreover, providing also provenance meta-data as RDF and making it available on the Web of Data [23], offers more interchange possibilities and transparency. This would let people link to provenance information from other sources. It provides them the opportunity to compare these sources and choose the most appropriate one or the one with higher quality.

In our specific context of DBpedia for example, by indicating by whom and when a triple was created (or contributed by), it could let any application flag, reject or approve this statement based on particular criteria (see Section 5).

In this paper we propose a modelling solution to semantically represent information about provenance of data in DBpedia and an extraction framework capable of computing provenance for DBpedia statements using Wikipedia edits. In particular, in the next section we overview some related work in the realm of provenance management on the Web of Data and in trust and quality evaluation techniques on wikis. Comparing these two research fields we highlight the limitations that we found in both of them: the former lacks of concrete and well-established procedures to support the integration and publication of provenance of non- or semi-structured data on the Web of Data; the latter does not take into account the importance of making the information generated analysing users’ edits available as Linked Data and providing details of the steps involved in the analysis. Then, in Section 3, we give some background information regarding lightweight ontologies such as SIOC (Semantically-Interlinked Online Communities) and its extensions which will be used in our modelling solutions. We decided to use the SIOC vocabulary and its extensions because it aims at describing the structure of online communities such as in wikis, and its Actions module suits well our need of defining events and user activities in wikis. In Section 4, we detail the W7 model for provenance representation, as previously designed by S. Ram et al. [33], and our implementation of this model with a lightweight ontology built to express it in RDFS. In the same section we also provide an alignment of our model with the Open Provenance Model (OPM), the reference ontology chosen by the W3C Provenance Incubator Group. Finally, in Section 5, we describe the architecture of our DBpedia provenance extraction framework. Then we detail how we model provenance information for DBpedia statements and expose it as Linked Open Data. Before concluding we also show a set of scripts to directly browse information about the statements on the DBpedia pages.

2. Related Work

Extracting and representing provenance information about data is a research topic that is going on from many years. Many studies have been conducted for representing provenance of data so far. Among all in [10] and [36] the authors provide comprehensive surveys about data provenance methodologies. The first one provides one of the first surveys in the realm of provenance management on the Web of Data and in trust and quality evaluation techniques on wikis. Comparing these two research fields we highlight the limitations that we found in both of them: the former lacks of concrete and well-established procedures to support the integration and publication of provenance of non- or semi-structured data on the Web of Data; the latter does not take into account the importance of making the information generated analysing users’ edits available as Linked Data and providing details of the steps involved in the analysis. Then, in Section 3, we give some background information regarding lightweight ontologies such as SIOC (Semantically-Interlinked Online Communities) and its extensions which will be used in our modelling solutions. We decided to use the SIOC vocabulary and its extensions because it aims at describing the structure of online communities such as in wikis, and its Actions module suits well our need of defining events and user activities in wikis. In Section 4, we detail the W7 model for provenance representation, as previously designed by S. Ram et al. [33], and our implementation of this model with a lightweight ontology built to express it in RDFS. In the same section we also provide an alignment of our model with the Open Provenance Model (OPM), the reference ontology chosen by the W3C Provenance Incubator Group. Finally, in Section 5, we describe the architecture of our DBpedia provenance extraction framework. Then we detail how we model provenance information for DBpedia statements and expose it as Linked Open Data. Before concluding we also show a set of scripts to directly browse information about the statements on the DBpedia pages.
[11]. Considering also existing semantic models for provenance of data, what is common between most of the modelling solutions is the presence of three concepts involved in the data life-cycle: Actors, Processes and Artefacts. Indeed, a modelling approach can be “process-oriented”, “data-oriented” (the two distinctions made in [36]), or “actor-oriented” (as proposed by [22]).

However these studies, and most of the studies about data provenance, are not focused on integrating provenance information into the Web of Data. In [23] the author explicitly addresses the characteristics of provenance of data from the Web, and proposes the “Provenance Vocabulary”\(^5\). We agree with the author on the fact that providing this information as RDF would make provenance metadata more transparent and interlinked with other sources, and it would also offer new scenarios on evaluating trust and data quality on the top of it. In this regard a W3C Provenance Incubator Group\(^6\) has been recently established. The mission of the group is to “provide a state-of-the art understanding and develop a roadmap in the area of provenance for Semantic Web technologies, development, and possible standardisation”. Requirements for provenance on the Web\(^7\), along with several use cases and technical requirements have been provided by the working group so far. These activities and documents have been recently included in a final report of the activities of the Incubator Group\(^8\). We invite the reader to consult this document in order to have more detailed information about the requirements for provenance needed in this work. In particular the requirements belonging to the following “dimensions”: object, attribution, process, versioning, publication and scale. The report contains also mappings between the most relevant provenance ontologies. Many ontologies representing provenance of data are taken into consideration (such as the Provenance Vocabulary, the Provenir ontology, the Open Provenance Model (OPM), etc.), as well as other lightweight ontologies (such as the Dublin Core\(^9\) vocabulary) that can partially represent provenance aspects of Web data. An alignment of these ontologies is provided in the aforementioned W3C document, and the model taken as reference for the mappings is the OPM (more details later in Section 4).

Finally, a comprehensive analysis of approaches and methodologies for publishing and consuming provenance metadata on the Web is exposed in [24].

Another research topic relevant to our work is the evaluation of trust and data quality in wikis. Recent studies proposed several different algorithms for wikis that would automatically calculate users’ contributions and evaluate their quantity and quality in order to study the authors’ behaviour, produce trust measures of the articles and find experts. WikiTrust [2] is a project aimed at measuring the quality of author contributions on Wikipedia. They developed a tool that computes the origin and author of every word on a wiki page, as well as “a measure of text trust that indicates the extent with which text has been revised”\(^10\). On the same topic other researchers tried to solve the problem of evaluating articles’ quality, not only examining quantitatively the users’ history\([27]\), but also using social network analysis techniques\([28]\). Another relevant contribution is in\([19]\), where the author details the implementation of a system for expert finding in Wikipedia.

From our perspective, there is a need of publishing provenance information as Linked Data from websites hosting a wide source of information (such as Wikipedia) and also from relevant datasets (such as DBpedia). Yet, most of the work on provenance of data is, either not focused on integrating provenance information on the Web of data, or mainly based on provenance for resource descriptions or already structured data. On the other hand, the interesting work done so far on analysing trust and quality on wikis does not take into account the importance of making the analysed data available on the Web of data.

Interesting and related research in our context is also presented in\([14]\) and\([15]\). First, the work by Vrandečić et al. describes a collaborative Web application that allows users to aggregate sources of information on entities of interest from the Web of Data. It takes Wikipedia as its starting point for its entities and it provides the source of every information added by its users. Then, the research presented by Ceolin et al. describes a trust algorithm for event data and an ontology representing events in general, the Simple Event Model. Interestingly the authors

\(^5\) http://purl.org/net/provenance/ns
\(^7\) http://www.w3.org/2005/Incubator/prov/wiki/User_Requirements
\(^8\) http://www.w3.org/2005/Incubator/prov/
\(^9\) http://wikitrust.soe.ucsc.edu/
\(^10\)http://wikitrust.soe.ucsc.edu/
provide a discussion of a mapping between OPM and the Simple Event Model using a similar methodology to ours (as we will detail in Section 4.2).

Overall it is important to mention a similar approach to the work in this paper that has been implemented and described in [12]. The authors propose an algorithm to compute trust values on Wikipedia articles using provenance information extracted from the revision history. The algorithm implemented to compute trustworthiness of assertions is based only on the internal links between articles and more specifically on citations. Hence this work is more focused on computing trust of Wikipedia articles rather than on representing and publishing provenance information to the Web of Data. A vocabulary for annotating the provenance information is used, it is called the Proof Markup Language (PML)\(^\text{11}\), but the data used by the experiment has not been published. However, since we focus on representing and publishing provenance of DBpedia to the Linked Open Data, we decided to use popular lightweight ontologies such as SIOC, Dublin Core and ChangeSet\(^\text{12}\) to represent edits in Wikipedia and changes to DBpedia statements. These popular ontologies have been integrated and extended with specific modelling solutions to represent more in depth the Wikipedia edits history (for more details see our W7 ontology implementation described in Section 4.1). Mappings to the OPM ontology have also been provided in order to facilitate the integration with other provenance data, as OPM has been chosen as a reference by the W3C Incubator Group (more details in Section 4.2). Furthermore with our work we show how we reused existing community ontologies and how these vocabularies can be applied to a concrete use case in order to represent provenance at a triple level and publish it as Linked Data.

3. Background

In the following section we provide an overview of the previous related work done by the authors. Furthermore we briefly describe popular lightweight ontologies, such as SIOC and its extensions, which will be used in our modelling solutions. We use the SIOC vocabulary because it aims at describing the structure of online communities such as in wikis, and its Actions module suits perfectly our need of defining

\(^{11}\)http://tw.rpi.edu/portal/Proof_Markup_Language
\(^{12}\)http://vocab.org/changeset/schema.html

events and user activities in wikis. The popularity of the SIOC vocabulary in the Linked Open Data cloud is also another important factor we took into consideration since our goal is to publish provenance to the Web of Data.

3.1. SIOC — Semantically-Interlinked Online Communities

The SIOC Ontology — Semantically-Interlinked Online Communities\(^\text{13}\) — provides a model for representing online communities and their contributions [13]. It is considered as one of the building blocks of the Social Semantic Web, since it is used in more than 50 applications, including the new release of Drupal 7\(^\text{14}\). It is mainly centred around the concepts of users, items (such as posts on a forum and wiki articles) and containers (such as online forums and wikis). Thus, it can be used to model content created by a particular user on several platforms, enabling a distributed perspective to the management of User-Generated Content on the Web.

In Figure 1\(^\text{15}\) the main concepts of the SIOC Core ontology and their relationships are displayed. As we can see from the picture the atomic elements of the Web applications described by SIOC are called Items. They are grouped in Containers, that can themselves be contained in other Containers. Finally, every Container belongs to a Space. Those abstract concepts are best understood when deepened into a concrete example, as the one given in the middle column: a Site may contain a number of Forums, some of them containing sub-forums, and

\(^{13}\)http://sioc-project.org
\(^{14}\)http://groups.drupal.org/node/16597
\(^{15}\)Figure from: http://sioc-project.org
every Forum contains a set of Posts. Every Post (and actually every Item, Container or Space) can be associated with Tags or Category/ies representing their topic. Moreover, we can see in that schema that SIOC also represents the UserAccounts creating Items. It is important to mention that the class UserAccount does not intend to represent the physical people using the application, but rather the online account they are using. In addition to the Core ontology, other modules (such as Types, Services and Access) are provided by the SIOC project in order to extend and refine its functionality and granularity. For more details about SIOC, we invite the reader to consult [9] and its online specification.

3.2. Using SIOC for wiki modelling

The SIOC Types module provides several subclasses of sioc:Container and sioc:Item which can be used to model wikis’ structure, including sioc:Wiki and sioc:WikiArticle. Basically a Wiki is a Container of WikiArticles, and each article is a specific timestamped revision which is linked to the UserAccount of its author with the sioc:has_creator property. However, looking more deeply into the SIOC modelling solution for wikis, we realised that some characteristics of wikis required further modelling. Hence, we extended the SIOC Ontology to take into account such characteristics [31]. We introduced new properties to model versioning in wikis, and we provided other modelling solutions to represent multi-authoring, different types of links, discussion pages, tags and categories. Then, to test the potential offered by our model, we developed some applications for exporting RDF data from popular wiki platforms and enabling search across different wiki systems [32]. In particular one of the SIOC RDF exporters is the “SIOC-MediaWiki Exporter”, a web service exporting RDF with our model from every wiki running on the MediaWiki platform. Later in Section 5 we will discuss how we used this exporter to generate RDF data for all the revisions of a collection of Wikipedia articles.

3.3. The SIOC Actions module

While SIOC represents the state of a community at a given time, SIOC-actions [17] can be used to represent their dynamics, i.e. how they evolve. Hence, SIOC provides a document-centric view of online communities and SIOC-actions focuses on an action-centric view. More precisely, the evolution of an online community is represented as a set of Actions, performed by a user with its UserAccount, at a specific time, and impacting a number of objects. Besides the SIOC ontology, SIOC-actions relies on the vocabulary for Linking Open Descriptions of Events (LODE) described in [35]. The core of the module is the Action class, which is a timestamped event involving an agent (typically a foaf:Agent) and a number of digital artefacts (class sioca:DigitalArtifact). Figure 2 displays a diagram with two representations of an Action linked to its timestamp and its actor.

The Action class is subclass of Event from the the Event Ontology. SIOC-actions provides an extensible hierarchy of properties for representing the effect of an action on its artefacts, such as creates, modifies, deletes, uses, etc. For a more detailed description of the implementation of SIOC Actions in a concrete example such as wikis, we invite the reader to consult Section 4.1.

4. Representing data provenance in RDFS/OWL

4.1. The W7 model

The W7 model is an ontological model created to describe the semantics of data provenance [33]. It is a conceptual model and, to the best of our knowledge, no RDFS/OWL representation of this model has been implemented yet. Hence in this paper we focus on an RDFS/OWL implementation of W7 for the specific context of wikis. As a comparison, in their previous work [34] Ram S. and Liu J. use Wikipedia as an example to theoretically illustrate how their proposed W7 model can capture domain or application specific provenance. Starting from the suggestions and the examples given by these authors we implemented the model described in their paper.

\[\text{http://rdfs.org/sioc/spec/}\]
\[\text{http://linkedevents.org/ontology/}\]
\[\text{Please note that the class sioc:User has been recently renamed in sioc:UserAccount}\]
The W7 model is based on the Bunge’s Ontology [16]. In other words, it is built on the concept of tracking the history of the events affecting the status of things during their life cycle. In this particular case we focus on the data life cycle. The Bunge’s ontology, developed in 1977 by Mario Bunge, is considered as one of the main sources of constructs to semantically model real systems and information systems. While Bunge’s work is mainly theoretical, there has been some effort from the scientific community to translate his work into machine readable ontologies 20. The W7 model can then be seen as an extraction of a part of the constructs described by the Bunge’s theories.

The W7 model represents data provenance using seven fundamental elements or interrogative words: what, when, where, how, who, which, and why. Hence very similar to the well-known “Five Ws” theory commonly practiced in journalism [20]. All the six interrogative words in the “Five Ws” theory are included in the W7 model. The seventh added word in the W7 model is which. In order to generate complete provenance information about a data source, it is necessary to provide an answer to all the seven questions. This model has been purposely built with general and extensible principles, hence it is possible to capture provenance semantics for data in different domains. We refer to [33] for a detailed description of the mappings between the W7 and Bunge’s models, and in Table 1 we provide a summary of the W7 elements (as in [34]).

Having described the structure of the SIOC Actions module in Section 3.3, and looking at the W7 model summarised in Table 1, it is clear the reason why we chose SIOC Actions as core of our model: most of the concepts in the Actions module are the same as in the W7 model; moreover wikis are community sites and the Actions module has been implemented to represent dynamic, action-centric views of online communities.

<table>
<thead>
<tr>
<th>Provenance element</th>
<th>Construct in Bunge’s ontology</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td>Event</td>
<td>An event (i.e. change of state) that happens to data during its life time</td>
</tr>
<tr>
<td>How</td>
<td>Action</td>
<td>An action leading to the events. An event may occur, when it is acted upon by another thing, which is often a human or a software agent</td>
</tr>
<tr>
<td>When</td>
<td>Time</td>
<td>Time or more accurately the duration of an event</td>
</tr>
<tr>
<td>Where</td>
<td>Space</td>
<td>Locations associated with an event</td>
</tr>
<tr>
<td>Who</td>
<td>Agent</td>
<td>Agents including persons or organisations involved in an event</td>
</tr>
<tr>
<td>Which</td>
<td>Agent</td>
<td>Instruments or software programs used in the event</td>
</tr>
<tr>
<td>Why</td>
<td>-</td>
<td>Reasons that explain why an event occurred</td>
</tr>
</tbody>
</table>

Table 1: Definition of the 7 Ws by Rams and Liu J.

In the following sections we provide a detailed description of how we answered each of these seven questions in order to build provenance data from wikis.

4.1.1. What

The What element represents an event that affected data during its life cycle. It is a change of state and the core of the model. In this regard, there are three main events affecting data: creation, modification and deletion. In the context of wikis, each of them can appear: users can (1) add new sentences (or characters), (2) remove sequences of characters, or (3) modify characters by removing and then adding content in the same position of the article. In addition, in systems like Wikipedia, some other specific events can affect the data on the wiki, for example “quality assessment” or “change in access rights” of an article [34]; however, they can be expressed with the three broader type defined above.

Since (1) wikis commonly provide a versioning mechanism for their content and (2) every action on a wiki article leads to the generation of a new arti-

---

In this regard, we decided to model a revert as all the other edits, and not as a particular pattern. The distinction between a revert and other types of action can be yet identified, with an acceptable level of precision, by looking at the user comment entered when doing the revert, since most users add a related revert comment (the same filtering approach is implemented in [18] with acceptable results) 21.

Going further, and to represent provenance data for the action involved in each wiki edit, we modelled the diffs existing between pages. To model the differences calculated between subsequent revisions we created a lightweight Diff ontology, inspired by the Changeset vocabulary 22. Yet, instead of describing changes to RDF statements (which is the scope of Changeset), the Diff model aims at describing changes to plain text documents. 23 It provides a main class, the diff:Diff class, and six subclasses: SentenceUpdate, SentenceInsertion, SentenceDeletion and ReferenceUpdate, ReferenceInsertion, ReferenceDeletion, based on the previous How patterns.

The main Diff class represents all information about the change between two versions of a wiki page (see Fig. 3). The Diff’s properties subjectOfChange and objectOfChange point respectively to the version changed by this diff and to the newly created one. Details about the time and the creator of the change are provided respectively by dc:created and sioca:has_creator. Moreover, the comment about the change is provided by the diff:comment property with range rdfs:Literal. In Figure 3 we also display a Diff class linking to another Diff class. The latter represents one of the six Diff subclasses described earlier in this section. Since a single diff between two versions can be composed by several atomic changes (or “sub-diffs”), a Diff class can then point to several subclasses using the dc:hasPart property. Each Diff subclass can have maximum one TextBlock removed and one added: if it has both, then the type of change is an Update, otherwise the type would be an Insertion or a Deletion.

The TextBlock class is part of the Diff ontology and represents a sequence of characters added or removed in a specific position of a plain text document.

---

21 Note that we could also compare the n-1 and n+1 version of each page to identify if a change is a revert.

22 The Changeset schema: http://purl.org/vocab/changeset/schema#.

23 The Diff ontology is publicly available at: http://vocab.deri.ie/diff#.
It exposes the content itself of this sequence of characters (content) and a pointer to its position inside the document (lineNumber). It is important to precise that usually the document content is organised in sets of lines, as in wiki articles, but this class is generic enough to be reusable with other types of text organisation. To note also that each of the six subclasses of the Diff class inherit the properties defined for the parent class, but unfortunately this is not displayed in Figure 3 for space reasons.

With the model presented it is possible to address an important requirement for provenance: the reproducibility of a process. Starting from an older revision of a wiki article, just following the diffs between the newer revisions and the TextBlocks added or removed, it is possible to reconstruct the latest version of the article. This approach goes a step further than just storing the different data versions: it provides details of the entire process involved in the data life cycle.

4.1.3. When

The When element in W7 is equivalent to the Time element from Bunge’s ontology, and obviously refers to the time an event occurs, which is recorded in every wiki platform for every page edit. As depicted in Figure 3, each Diff class is linked to the timestamp of the event using the dc:created property. The same timestamp is also linked to each Diff subclass using the same property (not shown in Fig. 3 for space reasons). The time of the event is modelled with more detail in the Action element as shown in the following Listing 2.24.

```turtle
<http://vmuss06.deri.ie/actions#title=Dublin_Core&id=383055>
  dc:created "2010-08-21T06:36:17Z"^^http://www.w3.org/2001/XMLSchema#dateTime;
  lode:atTime [a time:Instant;
    time:inXSDDateTime "2010-08-21T06:36:17Z"^^http://www.w3.org/2001/XMLSchema#dateTime].
}; a sioca:Action.
```

Listing 2. Representing the ”When” element in Turtle syntax

In this context we consider actions to be instantaneous. As in [17] we track the instant that an action is taking effect on a wiki (i.e. when a wiki page is saved). Usually, this creation time is represented using dc:created. Another option provided by LODE [35] uses the lode:atTime property to link to a class representing a time interval or an instant.

4.1.4. Where

The Where element represents the online “Space” or the location associated with an event. In wikis, and in particular in Wikipedia, this is one of the most controversial elements of the W7 model. If the location of an article update might be considered as the location of the user when updating the

24 For the namespaces see: http://prefix.cc
content, then this information on the Wikipedia is not completely provided or accurate. Indeed we can extract this information only from the IP address of the anonymous users but not from all the users contributing on the Wikipedia. So at the moment our solution is to just keep track of the IP address of the anonymous users as we can see in SIOC UserAccount URIs like this: http://en.wikipedia.org/wiki/User:96.245.230.136. We can also link each UserAccount with the related IP address using the sioc:ip_address property.

4.1.6. Which

The Which element represents the programs or the instruments used in the event. In our particular case it is the software used in editing the event, which might be a bot or the wiki software used by the editor. Since there is not a direct and precise way to identify whether the edit has been made by an human or a bot, our model does not differentiate that. A naive method could be to look at the username and check if it contains the “bot” string.

4.1.7. Why

The Why element represents the reasons behind the event occurrence. On the Wikipedia it is defined by the justifications for a change inserted by a user in the “comment” field. This is not a mandatory field for the user when editing a wiki page but the Wikipedia guidelines recommend to fill-in this text field. We model the comment left by the user with a property diff:comment linking the diff:Diff class to the related rdfs:Literal.

4.2. Alignment with the Open Provenance Model

Our proposed modelling solution is a particular implementation specific to the context of wikis. It is important to note that several generic ontologies representing provenance information have been developed. The scope of these vocabularies is to provide general purpose structures and terminologies that describe provenance information across different sets of application domains. Depending on each specific domain is then possible to refine and integrate these generic models with more specific vocabularies. The benefits of using common popular ontologies for provenance are clearly the interoperability of the applications using and producing provenance data, and the easy exchange of data between different sources and domains. The W3C Provenance Incubator Group (see Section 2) has recently published a document containing mappings between the most relevant provenance on-
In this document the ontology taken as reference for the mappings is the Open Provenance Model (OPM) [30]. OPM describes data life cycles in terms of processes (events or “things” happening), artefacts (“things” involved in a process), and agents (entities controlling “things” happening). These three are kinds of nodes within a graph, where each edge denotes a causal relationship. Edges have named types depending on the kinds of node they relate:

- a process used an artefact;
- an artefact was generated by a process;
- an artefact was derived from another artefact;
- a process was triggered by another process;
- a process was controlled by an agent.

As described in the W3C document providing the mappings, the motivations for the choice of the OPM are: (I) it is a general and broad model that encompasses many aspects of provenance; (II) it already represents a community effort that spans several years and is still ongoing, already benefiting from many discussions, practical use, and several versions; (III) many groups are already undergoing efforts to map their vocabularies to OPM.

For these reasons, and in order to align to the W3C Incubator Group’s choice, we defined the ontology mappings between the OPM and our proposed model. Hence here we follow the same procedures used by the W3C Group. The mappings, summarised in Table 2, are expressed using the SKOS [29]. The SKOS mapping properties are closeMatch, exactMatch, broadMatch, narrowMatch and relatedMatch. These properties are used to state mapping (alignment) links between SKOS concepts in different concept schemes, where the links are inherent in the meaning of the linked concepts. In the table we also provide a column with RDFS alignment properties. By using RDFS for mappings we benefit of reasoning capabilities over the data in our triplestore, hence our local store can be queried using OPM-based queries (assuming that RDFS inference support is available in the store).

To better understand the defined mappings and the reasons behind our choices we refer to the diagram displayed in Figure 5. In the diagram we show an implementation of the two models under comparison in this section. The one on the left represents our proposed “SIOC-based” model while the other one on the right the OPM. To note that the same instances, represented with different classes between the two models, are depicted with the same colours. Moreover, some properties not strictly relevant in this context have been omitted for more clarity, only the terms under comparison between the two models are displayed. In the following part of this section more details about this diagram are disclosed.

As summarised in Table 2, the first analysed mapping is about the opm:Process class which represents one or more actions “performed on or caused by artefacts, and resulting in new artefacts. On the


26 The SKOS Reference: http://www.w3.org/TR/skos-reference/
other hand the `sioca:Action` is a timestamped event involving a user and a number of digital artefacts. Therefore we can define the `Action` class as more specific (narrower) than the `Process` one, since it is limited to a timestamped instant and to digital artefacts. As regards the artefacts indeed, in the OPM model they are defined as “immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer”. While in the SIOC Actions module only the concept of `Digital Artifact` is contemplated. Even though the definition of `sioca:DigitalArtifact` is broad and generic (i.e. “Anything that can be the object of an Action”), we see this concept as narrower than the OPM one because it is restricted to digital objects. To the list of the artefacts we also included other objects like `sioct:WikiArticle` and `diff:Diff`. These are the artefacts involved in our context of wikis, and obviously they are defined as narrower concepts of the `opm:Artifact` class. In Figure 5 the aforementioned artefacts are defined as subclasses of the `opm:Artifact` class.

An important element of the provenance dynamics is the `Agent` or the entity “acting as a catalyst of a process, enabling, facilitating, controlling, or acting its execution” (as defined in OPM). The agent in our case is the user that contributes to the data on the wiki through his/her user account. The `sioc:UserAccount` class is defined as the representation of the account with which the user created the `Action`. Hence this concept is only related to the `Agent` concept since the user and his/her account are two disjoint concepts. For the same reason the properties `opm:wasControlledBy` and `sioc:has_creator`, which link a process or an action to an agent or a user account, have a `skos:relatedMatch` assigned.

In OPM five causal relationships (also called arcs or edges) are recognised. The `wasDerivedFrom` property links an artefact to another artefact that was a cause of its existence. As regards the data in wikis we have the mechanism of different versions of the data that are sequentially created one after the other. Hence the SIOC properties interlinking subsequent revisions (previous_version and related next/latest_version) have the same causal meaning, but limited to a more specific context. The arc `opm:used` defines the relation between a `Process` and an `Artefact` that has been necessary in the completion of the process itself. The `Process` requires the existence of the `Artefact` to initiate/terminate. Two properties in the Actions module are related to this property: `sioca:uses` and `sioca:modifies`; the latter is a sub-property of the former which points to “a digital artefact involved by the action, existing before and after it”. Since in the two models the existential requirement is persistent, the SIOC term is narrower than the OPM one because of its limitation to digital artefacts. As regards the `sioca:modifies` property, its definition is: “a digital artefact significantly altered by the action”; in our case this property is used to link the `Action` to the latest version of a wiki article, the one with an alias name that does not change over the time (e.g. `http://en.wikipedia.org/wiki/Ireland`).

<table>
<thead>
<tr>
<th>Terms from our “SIOC-based” model (subject)</th>
<th>SKOS Mappings</th>
<th>RDFS Mappings</th>
<th>Terms from Reference Model (OPM) (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>sioca:Action</td>
<td>skos:broadMatch</td>
<td>rdfs:subClassOf</td>
<td>opm:Process</td>
</tr>
<tr>
<td>sioca:DigitalArtifact, sioc:WikiArticle, diff:Diff</td>
<td>skos:broadMatch, skos:broadMatch, skos:broadMatch</td>
<td>rdfs:subClassOf, rdfs:subClassOf, rdfs:subClassOf</td>
<td>opm:Artifact</td>
</tr>
<tr>
<td>sioc:UserAccount</td>
<td>skos:relatedMatch</td>
<td>—</td>
<td>opm:Agent</td>
</tr>
<tr>
<td>sioc:previous_version</td>
<td>skos:broadMatch</td>
<td>rdfs:subPropertyOf</td>
<td>opm:wasDerivedFrom</td>
</tr>
<tr>
<td>sioca:uses, sioca:modifies</td>
<td>skos:broadMatch, skos:broadMatch</td>
<td>rdfs:subPropertyOf, rdfs:subPropertyOf</td>
<td>opm:used</td>
</tr>
<tr>
<td>(sioca:creates)</td>
<td>—</td>
<td>—</td>
<td>opm:wasGeneratedBy</td>
</tr>
<tr>
<td>sioc:has_creator</td>
<td>skos:relatedMatch</td>
<td>—</td>
<td>opm:wasControlledBy</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>opm:wasTriggeredBy</td>
</tr>
</tbody>
</table>

Table 2: Mappings between the Open Provenance Model and our proposed model based on SIOC terms.
On the other hand, each single revision\textsuperscript{27} is “created” (property sioca:creates) by the Action. This situation is closely matched by the OPM term wasGeneratedBy, but this does not have an alignment with the SIOCA term creates because they can be considered as inverse properties. To clarify, looking at the diagram in Figure 5, the three sioc:WikiArticle objects are (from the left to the right): the older modified revision of an article, the newer revision, and the latest alias version of the article that does not change URI.

Finally, in our model, we do not have a term that matches the opm:wasTriggeredBy term, which indicates that the start of a process was required for another process to be able to complete.

5. Extracting DBpedia provenance information

5.1. Overall approach

As we previously mentioned, our work was motivated by the need of delivering provenance information about DBpedia statements. According to DBpedia itself, its current dataset consists of 286 million triples (only for the English edition)\textsuperscript{28}. Associating provenance information to each of them could be relevant in several use-cases, especially for applications built on top of it. For example, by indicating by whom and when a triple was created (or contributed by), it could let any application flag, reject or approve this statement based on particular criteria. A site could decide to reject statements considered as being too new (so not having been checked by the page editor and the community), or because the author is not trusted in the area (e.g. the domain or range of the statement).

These needs for provenance management in DBpedia are even more relevant in the case of the upcoming DBpedia Live\textsuperscript{26} and the introduction of a new provenance element in the N-Quads DBpedia dump. This last feature is available only by downloading the N-Quads version of the DBpedia dump and it includes a provenance URI to each statement. The provenance URI denotes the origin of the extracted triple in Wikipedia by exposing the line and the section of a Wikipedia article where the statement has been extracted from. This is a first promising step that demonstrates the growth of interest in the topic. On the other hand with DBpedia Live, since information from Wikipedia will be immediately available in RDF and may be injected live in third party applications, it is important to provide this applications with means to decide if they should accept a statement or not. Finally, more than trustworthiness, provenance in DBpedia can be used for other purposes such as expert finding or social network analysis, focusing on the object-centred sociality vision, by identifying people contributing and socializing around similar resources. In both cases, more than resources, we could also rely on categories, that can be identified by selecting all resources associated to a particular DBpedia category, or more completely through SPARQL queries, such as identifying which people are contributing to pages about Web standards contributed by a particular organization.

To provide such features, we built a framework that

- on the one hand, extracts provenance information for DBpedia, using Wikipedia edits and
- on the other hand, makes that information available on the Web of Data, so that it can be used when building applications based on DBpedia.

We thus propose a twofold approach for provenance management from and for the Web of Data, combining Social Web paradigms (editing behaviours in Wikipedia) and Linked Data (provenance information about DBpedia in RDF). The system also makes Wikipedia edits available in RDF, letting Web Scientists interested in Wikipedia collaboration patterns getting relevant data using Semantic Web techniques and tools, rather than learning the Wikipedia API.

We will now describe the three elements of this framework. First, we show how we extract diff information from Wikipedia pages, whether it is individual pages, or pages under a common category. Then, we detail how provenance information about this page is extracted, and made available on the Web of Data using SCOVO — the Statistical Core Vocabulary\textsuperscript{29} [25] and the aforementioned Diff vocabulary. Finally, we describe how this information about Wikipedia pages is used to model provenance information regarding DBpedia statements, also available as RDF.

\textsuperscript{27}Each revision in Wikipedia has a URI that identifies the ID of the version, e.g.: http://en.wikipedia.org/w/index.php?title=Ireland&oldid=384683529

\textsuperscript{28}In DBpedia 3.6 as of Jan 2011: http://dbpedia.org/About

\textsuperscript{29}http://purl.org/NET/scovo#
5.2. Extracting and RDF-ising Wikipedia edits

The first step consists in collecting Wikipedia edits and building related diffs, as well as translating them into RDF. This information is used at a later stage to compute the provenance information, both in Wikipedia and DBpedia. To do so, we designed a script in order to get this information not only for a single page, but for a whole set of pages, belonging to the same category. Practically, the script:

- executes a SPARQL query on the DBpedia endpoint to get the subcategories of the seed one;
- stores these categories (hierarchically represented with SKOS) in a local triplestore;
- queries the DBpedia endpoint to identify all articles belonging to any of theses categories;
- generates (and stores locally) RDF data for each article using the SIOC-MediaWiki exporter that we previously built (Section 3);
- for each article looks recursively for the previous version and exports it in RDF.

Figure 6 describes the above steps involved in the whole provenance data collection process. Identifying pages in the same category can difficultly be done using only Wikipedia, and using DBpedia here (in combination with the former) provides a clear advantage.

Based on this dataset, a second script calculates and models the diff between all consecutive versions of the articles using the Wikipedia API. The API provides us HTML pages with the diff between two revisions, we need to parse these pages and then create the Diff objects modelled with the Diff vocabulary described previously in Section 4. Information about the editor, the timestamp, the comment and the ID of the versions collected at the previous step are merged with the diffs objects generated in this step. The script also identifies the type of change that happen between versions. This is done by comparing two consecutive versions to identify if the change was an Insertion or an Update or a Deletion. Then we identify if the change involved a reference or a normal sentence by parsing the content of the TextBlocks inside each Diff. That way, our export models changes not only as diff:Diff instances, but more precisely as Sentence or Reference Insertion/Update/Deletion. As for the previous extraction, all RDF information about the diffs is stored in the local triple-store, that hence contains all versioning and diff information about pages, modelled using SIOC, SIOC Types, SIOC Actions and the Diff vocabulary. Also, based on the mappings that we defined with OPM, this local store can be queried using OPM-based queries, providing that RDFS inference support is available in the store.

To evaluate this first step, we collected two datasets:

- a first one collecting all articles under the “Semantic Web” Wikipedia category (on the English Wikipedia) and all its subcategories;
- another one collecting all articles belonging to both “World Heritage Sites in Italy” and “Cities and towns in Emilia-Romagna” categories.

For that second one, we considered the intersection of the two groups of articles and consequently identified articles about World Heritage Sites in the Italian region Emilia-Romagna (for more details about this use-case see Section 5.4). Once again, this particular information cannot be directly retrieved from the Wikipedia articles, as the category does not exist, and has been obtained using a simple SPARQL query on DBpedia.

We also ran the diff extraction algorithm for the “Semantic Web” category. It generated data for all the 126 wiki articles belonging to this category and its subcategories recursively (9 categories in total). The total number of triples in the local triplestore for the “Semantic Web” use-case is almost 1.5 million triples, for a total of 8656 revisions.

5.3. Putting Wikipedia provenance back in the Web of Data

While our script collects and extract information from Wikipedia, it is only of limited interest in its current form. The second layer of our framework thus aims at making this information available on the Web (1) directly through Wikipedia pages and, (2) both for humans and machines. It thus can be used both by people browsing Wikipedia and that directly want to get an overview of the page (or the category) contributing users, or by agents that want to get statistics about these pages in a completely automated manner. The data stored in our triplestore is publicly available on the Web and accessible to software applications as RDF data directly using a RESTful web service.

The web service that provides raw RDF data is available at: http://vmuss06.deri.ie/WikiProvenance/index.php
sible to humans, is also based on the previous triple-
store. It consists in a Greasemonkey script \(^{31}\), which
identifies the Wikipedia page currently browsed and
sends this to a PHP script, which returns informa-
tion about the page, using SPARQL queries run on
the triplestore. This information is made available
on the top of each Wikipedia article, and exposes
information about the most active users on the arti-
cle and their edits. In addition, as we will see next,
this application also provides links to RDF repre-
sentation of this information available through our
web service. By being a Greasemonkey script, it can
be installed by anyone on Mozilla Firefox browsers
as well as other popular Web browsers supporting
it. This also imply that this information is not re-
stricted to RDF-savvy users (as if being in the RDF
store only), but can simply be browsed in the stan-
dard Wikipedia.

For each page, the script identifies the top con-
tributors (identified as the ones that made the most
edits), and computes for each of them:

- the total number of edits;
- the percentage of “ownership” on the page (i.e.
  the percentage of their edits compared to all
  the edits of the article);
- the number of lines added;
- the number of lines removed.
- the number of lines added or removed on all the

\(^{31}\)http://www.greasespot.net/
articles belonging to the category “Semantic Web”.

These information are then available as a table on the top of the page, as seen in Figure 7 (top figure) for the “Linked Data” page. For categories, similar information are identified, albeit identifying these statistics for all pages of the category, and not for a single page. Browsing a wiki category page, the application shows a list of the users with the biggest number of edits on the articles of the whole category (and related subcategories). It also shows the related percentages of their edits compared to the total edits on the category. It also exposes a list of the most edited articles in the category during the last three months. A screenshot of the result for categories can be seen in Figure 7 (bottom). We can also see, at the bottom of each table there is a link pointing to a page where a longer list of results will be displayed.

Furthermore, to make that information also available to machines, these statistics are made available in RDF. We especially relied on SCOVO\textsuperscript{32}, the Statistical Core Vocabulary\textsuperscript{[25]}. It relies on the concepts of Items and Dimensions to represent statistical information. In our context, the Item is one piece of statistical information (e.g. user “X” edited 10 lines on page “Y”) as displayed in the example in Listing 4. In a description of an Item various dimensions are involved:

- the type of information that we want to represent (number of edits, percentage, lines added and removed etc.);
- the URI of the page or the category impacted;
- the URI of the user involved.

Hence, we created four instances of scv:Dimension to represent the first dimension (as in Listing 3), and relied then simply on the scv:dimension property for the other ones. One issue yet with this approach is that it does not differentiate the dimension related to the user and the one related to the page, which is a limitation of SCOVO itself\textsuperscript{33}. In the future we may either create new properties, or check other vocabularies for representing statistics on the Web of Data such as the RDF Data Cube vocabulary\textsuperscript{34}, and SDMX\textsuperscript{35}. As an example, Listing 4 represents that the user KingsleyIdehen made 11 edits on the SIOC page.

With this single script, one can get the same information displayed using the Greasemonkey script and also to have the raw RDF description of the page requested. These scripts (the extraction framework and the provenance visualisers) are available at http://vmuss06.deri.ie/WikiProvenance/index.php, as well as the browser plug-ins. Also, a short video demonstrating the application is available at the address http://vmuss06.deri.ie/WikiProvenance/video/.

5.4. Modelling DBpedia provenance

Finally, the last step of the framework is to track provenance about DBpedia resources and statements, based on Wikipedia provenance infor-

\textsuperscript{32}http://purl.org/NET/scovo#

\textsuperscript{33}We considered using sioc:has_creator but semantically is not exactly the same, as the user is not creating the scovo:Item per se, but is just a part of its statistical information.

\textsuperscript{34}http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html

\textsuperscript{35}http://sdmx.org/
mation. Indeed, our goal is not only to provide provenance data from Wikipedia, but also to keep track of the changes happened in Wikipedia and identify what are the effects of these changes on the DBpedia dataset. In this section we show how we identify the authors of the triples stored in the DBpedia dataset and how we can relate them to the provenance details previously generated from the corresponding Wikipedia articles. The built application leverages the provenance data created for Wikipedia and combines it with the DBpedia extraction procedures. In order to retrieve the set of properties mapped from the infobox properties on Wikipedia to DBpedia, we took the mappings defined on the related DBpedia wiki.

In this wiki is possible to find the infobox-to-ontology and the table-to-ontology mappings which are used by the DBpedia extraction framework. The framework collects the templates defined in the wiki and extracts the Wikipedia content according to them.

As described in Section 5.2 for our specific use case about the “World Heritage Sites in Emilia-Romagna” we collected pages belonging to two different categories. All the articles resulting from the intersection of the two categories use one particular Wikipedia Infobox called “Infobox Italian comune”. This “table template” defines the properties associated with all the articles about cities in Italy. The structure of this template is shown in Listing 5, where part of the Infobox source text of the article “Modena” is displayed. The wiki text displayed is then translated and rendered by Wikipedia in a table usually on the top right corner of the page.

```
{{Infobox Italian comune
| name = Modena
| official_name = Comune di Modena
| native_name = [...]
| postal_code = 41100
| area_code = 059
| website = {{official|http://www.comune.modena.it}}
| footnotes = }}
```

Listing 5. An excerpt of the Wikipedia “Infobox Italian comune” from the article “Modena”

Once the mappings between Wikipedia and DBpedia were retrieved and the provenance data for the Wikipedia articles generated and stored using our data extraction framework, our application was ready to be implemented. A PHP script has been developed to analyse the content of the TextBlocks of
each Diff stored in our dataset (Section 4.1). A single SPARQL query is necessary to get the content of the diffs which are probably related to some changes happened in the Infobox part of the wiki article. The aforementioned query is displayed in Listing 6. For each change happened in the first 30 lines of the article’s revisions it returns the user, the timestamp, the object of change, the content of the line changed and the position of the line in the article. The reason for the line number restriction is because, in our case, the Infobox properties are always positioned in this part of the articles.

```
SELECT distinct ?user ?date ?obj ?content ?line WHERE {
  GRAPH <cities> {
    ?diff rdf:type diff:Diff ;
    dct:hasPart ?subdiff ;
    dc:created ?date ;
    sioc:has_creator ?user ;
    diff:objectOfChange ?obj ;
    FILTER (regex (?obj , "$.pagetitle.")).
    ?txtblk rdf:type diff:TextBlock ;
    diff:content ?content ;
    diff:lineNumber ?line .
    FILTER (?line < 30).
  }
}
```

Listing 6. A SPARQL query to retrieve the lines changed between all the revisions of an article. Line numbers should be less than 30.

Also note that in Listing 6 the title of the article is represented by the PHP variable $pagetitle.

The application then analyses each line content returned by the query to identify the changes that actually involved the Infobox properties. For each of the changes matching the requirements, their details (user, timestamp, page version, etc.) and the related DBpedia property affected by the change, are stored again in the local triplestore. The results are semantically modelled using the SIOC Actions-based model previously described in Section 4.1. The only difference here is the use of the Changeset vocabulary[^37] to model the changes of the DBpedia triples caused by the Wikipedia Infobox modifications. As described in Section 4.1 the Changeset protocol [1] is similar to the Diff model we adopted in this work. Instead of having a Diff class that points to added or removed TextBlocks, the Changeset vocabulary defines a cs:ChangeSet class that points to the resources subject and object of change and to the rdf:Statements added and removed. Each Statement is then composed by one rdf:subject, one rdf:predicate and one rdf:object. Similarly to what previously described, a sioca:Action is then linked to a cs:ChangeSet with the property sioca:creates. In Listing 7 we show a modelling example of a ChangeSet in DBpedia.

```
<http://vmuss06.deri.ie/actions#title=Modena&id=383055> [...]
sioca:creates <http://vmuss06.deri.ie/changesets#title=Modena&prop=province&date=2009-10-09T04:38:53Z>;
  a sioca:Action .
<http://vmuss06.deri.ie/changesets#title=Modena&prop=province&date=2009-10-09T04:38:53Z>
  sioc:has_creator <http://en.wikipedia/User:Plasticspork>;
  cs:changeReason "Change in Wikipedia";
  cs:createdDate "2009-10-09T04:38:53Z";
  cs:subjectOfChange <http://dbpedia.org/resource/Modena>;
  cs:addition _:bnode1;
  cs:removal _:bnode2;
  rdfs:seeAlso <http://vmuss06.deri.ie/DBpediaStats#title=Modena&prop=province&date=2009-10-09T04:38:53Z#edits>
  rdfs:seeAlso <http://vmuss06.deri.ie/DBpediaStats#title=Modena&prop=province&date=2009-10-09T04:38:53Z#users>
  a cs:ChangeSet.
_:bnode1
  rdf:subject <http://dbpedia.org/resource/Modena>;
  rdf:predicate <http://dbpedia.org/ontology/province>;
  rdf:object "Province_of_Modena";
  a rdf:Statement.
_:bnode2
  rdf:subject <http://dbpedia.org/resource/Modena>;
  rdf:predicate <http://dbpedia.org/ontology/province>;
  rdf:object "Modena";
  a rdf:Statement.
```

Listing 7. A ChangeSet for the DBpedia resource “Modena” expressed in Turtle. The object of the property “province” has changed from “Modena” to “Province_of_Modena”.

Please note that, in Listing 7, the ChangeSet instance links with seeAlso properties to two resources providing statistical information in RDF about the dbpedia:province property. The first one is about the number of edits to this property, on
this page, at the time of this ChangeSet. And the second one is similar but with the difference that is about the number of users who edited the property. These statistics are modelled using the SCOVO vocabulary and the resources in this example are explained later in this section in Listing 8.

Once all the diffs have been analysed, and the related data loaded into the triplestore, we focused our attention on the final part of the application. It is composed by a Mozilla Greasemonkey script which loads a table on the top of the DBpedia pages based on the results retrieved by another PHP script. The structure of this part of the application is similar to the structure described in Section 5.3 for the Greasemonkey script running on Wikipedia pages. Similarly, the PHP script receives a request from the Greasemonkey script for a specific DBpedia resource, then it queries the triplestore and replies to the Greasemonkey script with the results embedded in a HTML table. A screenshot of the table displayed on a DBpedia page is shown in Figure 8.

In accordance to what we did for the Wikipedia provenance data (Section 5.3), to make this information about DBpedia also available to machines, we provide these statistics in RDF. Using the SCOVO vocabulary we are able to model, for each property on each DBpedia page, the total number of edits and the number of users contributing to them. In the scv:Items implemented in this case the three dimensions involved are:
- the type of information that we want to represent (number of edits or number of users);
- the URI of the DBpedia resource impacted;
- the URI of the DBpedia property involved.

Hence, we created two instances of scv:Dimension to represent the first dimension (as in the first part of Listing 8). The other two dimensions are URIs linked with the scv:dimension property (second part of Listing 8).

```PREFIX dbst:<http://vmuss06.deri.ie/ DBpediaStats#>
PREFIX scv:<http://purl.org/NET/scovo#>

@prefix dbst:<http://vmuss06.deri.ie/ DBpediaStats#>
@prefix scv:<http://purl.org/NET/scovo#>

dbst:DBPropertyEdits rdf:s:subclassOf scv: Dimension;
dc:title "Number of edits for the property" .
dbst:Edits a :DBPropertyEdits;
dc:title "Number of edits" .
dbst:Users a :DBPropertyEdits;
dc:title "Number of users editing the property" .
```

---

dbst:title=Modena&prop=province&date=2009-10-09T04:38:53Z# edits a scovo:Item;
rdf:value 4;
scv:dimension dbst:Edits;
scv:dimension <http://dbpedia.org/resource/Modena>;

dbst:title=Modena&prop=province&date=2009-10-09T04:38:53Z# users a scovo:Item;
rdf:value 3;
scv:dimension dbst:Users;
scv:dimension <http://dbpedia.org/resource/Modena>;

Listing 8. Representing the number of edits and editors of the DBpedia properties with SCOVO

To give a clearer picture of the amount of data generated for this test, we now provide some technical details about the experiment conducted. The total amount of RDF triples generated and stored in our RDF-store is around 770.000. This includes all the provenance data about three Wikipedia articles (“Modena”, “Ferrara” and “Ravenna”) and other data about the structure of the two categories “World Heritage Sites in Italy” and “Cities and towns in Emilia-Romagna” and their members. The total number of members belonging to these two categories and all the subcategories is 2645 articles, but for these we did not collect all the revisions, we did that only for the intersection of the two categories. As regards the number of revisions of the three articles collected, each of them has almost 500 revisions.

In terms of time spent for the data acquisition process on a basic single core machine, the total process took around five hours:
- around three hours to get the data from DBpedia and the SIOC-MediaWiki exporter (the slowest part of the acquisition process because of the high number of requests to the Wikipedia API);
- two hours to get all the diffs between the revisions;
- a few minutes to analyse the diffs and match the DBpedia properties.

To better estimate the amount of RDF triples that can be generated by this process, we now provide a comparison between the DBpedia dataset and the result of our provenance extraction process applied
to the whole English Wikipedia. In September 2010 the English Wikipedia hosted around 3.5 million articles, with an average number of revisions per article equal to 73.5\textsuperscript{38}. Therefore we approximately consider a total of 257.25 million revisions. Since with our experiment we generated around 50.98 statements per revision, then for the whole English Wikipedia corpus we would generate almost 13.115 billion RDF triples. Considering that one part of all these statements describes the content and structure of the revisions, and the other part aims at describing their provenance, we estimate that the whole Wikipedia provenance dataset would consist of approximately 7 billion triples. As a comparison, the DBpedia dataset\textsuperscript{39} consists of 672 million RDF triples out of which 286 million were extracted from the English edition of Wikipedia and 386 million were extracted from other language editions and links to external datasets.

6. Conclusions and Future Work

In this paper we provided a solution for generating provenance information on DBpedia starting from the representation of provenance of data in Wikipedia. In particular our contributions include a specific lightweight ontology for provenance in wikis, based on the W7 model, and an alternative modelling solution based on the Open Provenance Model (with ontology alignments between both). Then we developed a framework for the extraction of provenance information for DBpedia data, using information from the Wikipedia revision history. Two applications browsing the generated data in a user-friendly and meaningful way on Wikipedia and DBpedia have been deployed. These applications are also capable of exposing the generated data, and statistics about the data itself, to the Web of Data. Finally, by combining fine-grained provenance data from Wikipedia and DBpedia we were able to provide authorship and versioning information for DBpedia RDF triples.

The W7 model that we used to model provenance in wikis, and Wikipedia in particular, is generic enough to be compliant with our modelling requirements. Clearly, it needs to be refined and specialised

\textsuperscript{38}From Wikipedia statistics hosted by the Wikimedia Foundation (September 2010): http://stats.wikimedia.org/EN/TablesWikipediaEN.htm

\textsuperscript{39}DBpedia dataset version 3.6, officially released in January 2011: http://blog.dbpedia.org/2011/01/17/dbpedia-36-released/
depending on the domain and use-case, but this approach is also valid for every other more detailed provenance ontology. Even the Open Provenance Model, with which we provided an alternative modelling solution, has to be specialised reusing or introducing other classes or properties.

With our experiments, we also demonstrated that, even using a lightweight ontology, the amount of triples required to represent provenance information could be really important. This is especially the case when capturing information about each version of highly-changing data, as wikis. As an estimation, assuming that the number of the triples grows linearly with the number of revisions collected, running our experiment for provenance extraction and representation with the whole English Wikipedia would generate 7 billion RDF triples. It shows the need for scalable RDF stores, and maybe as well for some other approaches to manage provenance information, such as Named Graph\cite{37} (as we have done in previous work on Semantic Wikis) or Annotated RDF, which would however require non-standard techniques for storing and using this provenance information. The same issue is highlighted by the W3C Incubator Group in the Provenance XG Final Report published in December 2010, where (in Section 5, “State of the Art and Technology Gaps”) the lack of “proven approaches to manage the scale of the provenance records to be recorded and processed” is stated\footnote{http://www.w3.org/2005/Incubator/prov/XGR-prov-20101214/}.

Future developments will include improvements and extensions of the potentialities of our application. We plan to increase the level of accuracy in matching the statements changed between revisions: at the moment we are still having some errors in matching the triples changed when we parse the content of the diffs. Moreover, we plan to extend the number of properties identifiable on a page by extending the number of mappings between Wikipedia and DBpedia. Currently our mappings correspond to the list of properties included in the Infoboxes, however DBpedia extracts other statements from other parts of the Wikipedia articles. These additional mappings can be probably found looking more deeply at the DBpedia Extraction Framework\footnote{http://wiki.dbpedia.org/Documentation}. An important improvement to our model will also be a provenance representation of the DBpedia extraction process itself, in a way that the activities that were carried out by the DBpedia Framework to generate or access the content could be recorded.

A new interesting feature for our system might be a new architecture that extracts the data for all the revisions from the Wikipedia dumps, and then updates this data with information from the newer revisions in real-time. It would be possible to follow the Wikipedia recent changes and update the data as soon as it changes on the wiki. This option would make us save time in collecting all the revisions of the articles because of the possible delays or errors happening using the Wikipedia API. The same approach can also be implemented using the new DBpedia Live service (described in Section 5.4) which updates in real-time its dataset as soon as the Wikipedia pages change.

It would be also interesting to investigate more new possible applications built on top of the provenance data presented here. Especially in the expert finding and user-profiling research areas, where we can imagine mining user profiles according to the users’ activity. We can evaluate not only the quantity of the users’ edits but also measure the quality of their contributions. This would be possible thanks to the fine granularity of the data we generate. Thus, on top of these user profiles we can also imagine to run expert finding algorithms, and also integrate these profiles with other social platforms. Based on the expertise level of the authors, the number of contributions over time, or the last time the statements were modified, we can have a system that decides which are the statements that we can trust and which are the ones we distrust. Extending this approach to other sources of Linked Data, in some situations we would also have the ability to choose between two or more data sources (e.g. DBpedia and Freebase\footnote{http://www.freebase.com/}) depending on which one provides more trustworthy values for particular properties. In this way our system goes in the same direction of the Tim Berners-Lee’s idea about the “Oh, yeah?” button\footnote{Quoting from http://www.w3.org/DesignIssues/UI.html, 1997}: “At the toolbar (menu, whatever) associated with a document there is a button marked “Oh, yeah?”. You press it when you loses that feeling of trust. It says to the Web, “so how do I know I can trust this information?”. The software then goes directly or indirectly back to metainformation about the document, which suggests a number of reasons.”.
References

[36] Y. Simmhan, B. Plale, D. Gannon, A survey of data provenance techniques, Computer Science Department, Indiana University, Bloomington IN 47405.