Smushing RDF instances: are Alice and Bob the same open source developer?

Lian Shi[1], Diego Berrueta[1], Sergio Fernández[1],
Luis Polo[1], and Silvino Fernández[2]

[1] Fundación CTIC
Gijón, Asturias, Spain
{firstname.lastname}@fundacionctic.org
http://www.fundacionctic.org/
[2] R&D Technological Centre (CDT)
ArcelorMittal Asturias
Avilés, Asturias, Spain
silvino.fernandez@arcelormittal.com
http://www.arcelormittal.com/

Abstract. Analysing RDF data gathered from heterogeneous Semantic Web sources requires a previous step of consolidation in order to remove redundant instances (data smushing). Our aim is to explore and integrate smushing techniques to improve recall, i.e., to find as many redundant instances as possible. Two approaches to spot resources with the same identity are described: the first one is based on Logics, exploiting OWL inverse functional properties (IFP); the second one is based on traditional IR techniques, e.g., resource label comparison. We evaluate experimental results in the context of open source communities.

1 Introduction

The increasing amount of machine processable data in the Semantic Web facilitates processes such as social network analysis and data mining. Innovative applications, like expert finding on the (Semantic) Web, are enabled by the ability of executing these processes at a World-Wide Web scale. Although the RDF data model is well suited to seamlessly merge data (triples) from arbitrary sources, a data integration problem still remains. Unconnected descriptions of the same thing can be obtained from different sources. For instance, a single individual can participate in several web communities with different virtual identities. When they are summed together, the descriptions of her virtual identities (such as e-mail accounts) will be different RDF resources weakly connected to each other. If these identities were to be taken as different persons, data analysis would be crippled, as it would lead to imprecise conclusions and a widespread flooding of phantom virtual identities.

Social communities, their networks and their collaborative forums, are one particular focus of interest for analysis, as they provide large amounts of data that can be used for several purposes. People in these communities share common interests, exchange information and interact with each other. FOAF [3] (short for Friend-Of-A-Friend) and SIOC [2] (Semantically-Interlinked Online Communities) offer vocabularies for publishing machine readable descriptions of people, making it possible to link from one site, person, company, etc. to related ones. With their popularity and wide acceptance as a de facto standard vocabularies for representing social networks, there is a dynamic increase in the amount of social profiles available in these formats produced by many large social networking websites. This fact can be verified by the number of documents that use these namespaces in the Semantic Web [6].

We use data mined from open source communities to evaluate two smushing techniques in order to merge the virtual identities of the members of these communities. That is, we aim to identify the co-occurrence of the same person in different communities[3]. The first approach exploits the semantics of inverse functional properties, which solely and definitely determines whether two entities are the same considering their property values. The second approach is not based on Logics, but on heuristics, more precisely, on the comparison of entity labels. Both techniques are applied to a dataset that contains thousands of instances of `foaf:Person`.

The paper is structured as follows: next, we briefly introduce the most important related work. We detail two complementary smushing strategies in Section 3. Section 4 describes how a corpus of RDF data was collected from some communities that focus on open source software development. Experimental results are exposed in Section 5, and their interpretation in the context of open source communities is discussed in Section 6. Finally, Section 7 closes the paper with conclusions on the smushing process and some insights into future work.

2 Related Work

Social networks have opened up a new sight because people provide information about themselves and their social connections in publicly accessible forums. The main topics and subjects of a vast literature of previous works about social networks include examinations of online social networks such as [10], which recommends a survey-based approach for extracting social information about users. Their growth and activity patterns, design and behaviour in online communities has also been studied [11, 20].

In [14], the authors show how large isolated data graphs from disparate structured data sources can be combined to form one, large, well-lined RDF graph. Their work provides a large corpus that can act as a benchmark dataset for evaluating expert finding algorithms, and it also simulates the availability of real-world data used in various research scenarios.

Ding et al. [5] present a novel perspective of the Semantic Web of FOAF documents, and proposed a heuristic approach to identify and discover FOAF

[3] In this sense our research relates to matching frameworks, see `http://esw.w3.org/topic/TaskForces/CommunityProjects/LinkingOpenData/EquivalenceMining`

documents from the Web and to extract information about people from these documents. Their work can be used to discover existing and emerging online communities.

The application of machine learning technologies to FOAF has also been explored, highlighting the challenges posed by the characteristics of such data. The authors of [12] experiment with profiles and generate a set of rules for adding properties to users found to be in a set of clusters, and also for learning descriptions of these groups. They argue these descriptions can be used later for on-the-fly personalisation tasks.

3 RDF data smushing

We call *smushing* to the process of normalising an RDF dataset in order to unify *a priori* different RDF resources which actually represent the same thing[4]. The application which executes a *data smushing* process is called a *smusher*. The process comprises two stages: first, redundant resources are identified; then, the dataset is updated to reflect the recently acquired knowledge. The latter is usually achieved by adding new triples to the model to relate the pairs of redundant resources. The `owl:sameAs` is often used for this purpose, although other properties without built-in logic interpretations can be used as well (e.g.: `ex:hasSimilarName`). We will expand on this at the end of this section.

Redundant resources can be spotted using a number of techniques. In this paper, we explore two of them: (1) using logic inference and (2) comparing labels. We note that other approaches are possible as well, including custom rule-based systems, human computation and user-contributed interlinking (UCI) [13].

3.1 Inverse Functional Properties

OWL [18] introduces a kind of object properties called *Inverse Functional Properties* (`owl:InverseFunctionalProperty`, IFPs for short). An IFP is a property which behaves as an injective association, hence its values uniquely identify the subject instance:

$$\forall p / p \in \text{IFP} \Rightarrow (\forall s_1, s_2 \; / \; p(s_1) = p(s_2) \Rightarrow s_1 = s_2) \qquad (1)$$

This inference rule is built-in in the OWL-DL reasoners, therefore, this kind of smushing can be easily achieved just by reasoning the model. However, it is advisable to avoid the reasoner and to implement the IFP semantics by means of an *ad hoc* rule. These are the reasons:

- Executing a simple, light-weight rule is often more efficient than the reasoner, which usually performs many other tasks. Moreover, it can be used regardless of the expressivity level of the dataset, while reasoners have unpredictable behaviour for OWL-Full datasets.

[4] This use of the expression *data smushing* in this context can be traced back to Dan Brickley: `http://lists.w3.org/Archives/Public/www-rdf-interest/2000Dec/0191.html`

```
CONSTRUCT {
  ?person1 owl:sameAs ?person2
}
WHERE {
  ?person1 rdf:type  foaf:Person .
  ?person2 rdf:type  foaf:Person .
  ?person1 foaf:mbox_sha1sum ?email .
  ?person2 foaf:mbox_sha1sum ?email .
  FILTER (?person1 != ?person2)
}
```

Fig. 1: IFP smushing rule implemented as a SPARQL CONSTRUCT sentence. The usual namespace prefixes are assumed.

– A custom rule can generalise IFPs to any kind of properties, including datatype properties. There are some scenarios in which such generalisation is useful. For instance, while the object property `foaf:mbox` is declared as an IFP, whose value is often unavailable due to privacy concerns. On the other hand, values of the property `foaf:mbox_sha1sum` are widely available (or can be easily calculated from the former), but as it is a datatype property, it cannot be declared an IFP in OWL.

This rule can be written as a SPARQL CONSTRUCT sentence, according to the idiom described by [19], see Figure 1. Note that this rule only takes into account the `foaf:mbox_sha1sum` property, but its generalisation to any property declared as `owl:InverseFunctionalProperty` is straightforward.

When smushing resources that describe people, some FOAF properties can be used as IFPs. The FOAF specification defines `mbox`, `jabberID`, `mbox_sha1sum`, `homepage`, `weblog`, `openid` as IFP, among others. However, a quick analysis of a set of FOAF files collected from the web shows that some of these properties are barely used, while others are often (mis-)used in a way that makes them useless as IFP. Notably, some users point their `homepage` to their company/university homepage, and `weblog` to a collective blog. Therefore, we restrict our smusher to the `mbox_sha1sum` property.

3.2 Label similarity

The concept of similarity is extensively studied in Computer Science, Psychology, Artificial Intelligence, and Linguistics literature. String similarity plays a major role in Information Retrieval. When smushing people's descriptions, labels are personal names (`foaf:name`). Nevertheless, personal names follow particular rules which make them intractable. Personal names, as any other word, can be miss-spelled, however, the probability of such error is very low if the names are entered by their owners. Therefore, traditional similarity comparison functions, such as Levenshtein distance, are not really useful. In [4] the authors describe advanced techniques for personal name comparison, and in [17] study their impact on the precision. We just use a much simpler strict string equality comparision, ignoring common invalid names often found in email headers.

Smushing based on label similarity deals with imprecise knowledge, i.e., even a perfect label equality does not guarantee that two resources are the same. Using a softer comparison function will produce even more uncertain knowledge.

A label-based smusher can be implemented as a rule. Unfortunately, SPARQL does not have rich built-in string comparison functions. There is a proposed extension call iSPARQL [16] that can be used with this purpose. Our experience reveals that iSPARQL implementation is far from being efficient enough to deal with large datasets. This fact suggests that other approaches to implement label-based smushing should be considered.

3.3 Smushing, correctness and consistence

The pairs of redundant resources identified using the techniques described above can be used to enrich the dataset. OWL provides a special property to "merge" identical resources, `owl:sameAs`. When two resources are related by `owl:sameAs`, they effectively behave as a single resource for all the OWL-aware applications. Note, however, that plain SPARQL queries operate at the RDF level, and therefore they are unaware of the `owl:sameAs` semantics.

Anyway, the semantics of `owl:sameAs` may be too strong for some cases. On the one hand, some applications may still want to access the resources individually. On the other hand, several factors can influence on the reliability of the findings made by the smusher. Notably, the data smushing based on label comparison is obviously imperfect, and can lead to incorrect results. For instance, different people can have the same name, or they can fake their identities. Even the logically-sound smushing based on IFPs is prone to error, due to the low-quality of the input data (fake e-mail addresses, identity theft). Although improbable, it is also possible that different e-mail addresses clash when they are hashed using SHA1 [7].

To tackle these issues, a custom property can be used instead, such as `ex:similarNameTo`. Applications interested in the strong semantics of `owl:sameAs` can still use a rule to re-create the links.

Another kind of OWL properties, Functional Properties (FP) are also useful for smushing. They can help to check the consistency of the smusher's conclusions. A resource cannot have multiple different values for a FP. Therefore, if two resources that are to be smushed are found to have irreconcilable values for `foaf:birthday`, an issue with the smushing rules (or the quality of the input data) must be flagged.

4 Data recollection

A corpus of RDF data with many `foaf:Person` instances was assembled by crawling and scrapping five online communities. There is a shared topic in these communities, namely open source development, hence we expect them to have a significative number of people in common.

We continue the work started in [1] to mine online discussion communities, and we extend it to new information sources. We use the following sources:

- GNOME Desktop mailings lists: all the authors of messages in four mailing lists (*evolution-hackers*, *gnome-accessibility-devel*, *gtk-devel* and *xml*) within the date range July 1998 to June 2008 were exported to RDF using SWAML [9].
- Debian mailing lists: all the authors of messages in four mailing lists (*debian-devel*, *debian-gtk-gnome*, *debian-java* and *debian-user*) during years 2005 and 2006 were scrapped from the HTML versions of the archives with a set of XSLT style sheets to produce RDF triples.
- Advogato: this community exports its data as FOAF files. We used an RDF crawler starting at Miguel de Icaza's profile. Although Advogato claims to have +13,000 registered users, only +4,000 were found by the crawler.
- Ohloh: the RDFohloh project [8] exposes the information from this directory of open source projects and developers as Linked Data. Due to API usage restrictions, we could only get data about the +12,000 oldest user accounts.
- Debian packages: descriptions of Debian packages maintainers were extracted from *APT* database of Debian packages in the *main* section of the *unstable* distribution[5].

Instances generated from these data sources were assigned a URI in a different namespace for each source. Some of these data sources do not directly produce instances of `foaf:Person`, but just instances of `sioc:User`.An assumption is made that there is a `foaf:Person` instance for each `sioc:User`, with the same e-mail address and name. These instances were automatically created when missing. This assumption obviously leads to redundant instances of `foaf:Person` which will be later detected by the smusher.

5 Experimental Results

The ultimate goal of our experiments is to exercise the smushing processes described in Section 3 against a realistic dataset. Two million RDF triples were extracted from the sources described above, and put into OpenLink Virtuoso server[6] which provides not only an effective triple store, but also a SPARQL end-point that was used to execute queries using scripts. Table 1a summarises the number of instances of `foaf:Person` initially obtained from each source.

We evaluated two smushers: the first one smushed `foaf:Person` instances assuming that `foaf:mbox_sha1sum` is an IFP; the second one smushed the same instances comparing their `foaf:name` labels for string strict equality, without any

[5] Retrieved August 3rd, 2008.

[6] `http://virtuoso.openlinksw.com/`

Source	`foaf:Person` instances
DebianPkgs	1,845
Advogato	4,168
GnomeML	5,797
Ohloh	12,613
DebianML	12,705

(a)

	Num. of people
In 5 communities	1
In 4 communities	37
In 3 communities	273
In 2 communities	1,669

(b)

Table 1: Size of the studied communities before smushing (1a), and people accounting by the number of communities they are present in, after smushing (1b).

normalisation. Both smushers were implemented using SPARQL CONSTRUCT rules. The newly created `owl:sameAs` triples were put in different named graphs.

These links were analysed to find co-occurrences of people in different communities. The absolute co-occurrence figures are presented in Tables 2 and 3, respectively. Later, the two named graphs were aggregated. Table 4 contains the absolute co-occurrence considering the combined results of both smushers. Note that some redundancies are detected by both smushers, therefore figures in Table 4 are not the sum of two previous ones. These matrices are symmetrical, hence we skip the lower triangle.

The degree of overlap between communities is better observed in Tables 5, 6 and 7, which present the ratio of overlap relative to the size of each community.

Tables 1b and 8 study the number of communities each person is present in. Interestingly enough, an individual was found to have presence in all five communities.

6 Discussion

The elements of the main diagonal of Tables 2, 3, 4 show the overlap within each community, i.e., the number of people that have registered more than once in each community. Some communities use the e-mail address as the primary key to identify their users, therefore, the smushing process using the e-mail as IFP (Table 2) has zeros in the main diagonal for these communities. However, other communities use a different primary key, thus allowing users to repeat their e-mail addresses. For instnace, a small number of users have registered more than one account in Advogato with the same e-mail (these account have been manually reviewed, and they seem to be accounts created for testing purposes).

Our data acquisition process introduces a key difference between how user accounts are interpreted in Debian mailing lists and GNOME mailing lists. The former considers e-mail address as globally unique, i.e., the same e-mail address posting in different Debian mailing lists is assumed to belong to the same user. On the other hand, a more strict interpretation of how Mailman works is made with respect to the GNOME mailing lists, where identical e-mail address posting in different mailing lists are assumed to belong to a priori different users. In the

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	0	81	37	74	762
Advogato		19	270	106	141
GnomeML			364	112	161
Ohloh				0	115
DebianML					0

Table 2: Number of smushed instances of `foaf:Person` using `foaf:mbox_sha1sum` as IFP.

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	98	170	101	58	1319
Advogato		49	592	95	305
GnomeML			1716	148	432
Ohloh				13	208
DebianML					2909

Table 3: Number of smushed instances of `foaf:Person` with exactly the same `foaf:name`.

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	98	188	113	104	1418
Advogato		55	669	167	342
GnomeML			1765	227	462
Ohloh				13	287
DebianML					2909

Table 4: Number of smushed instances of `foaf:Person` combining the two smushing techniques.

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	0.00%	4.39%	2.01%	4.01%	41.30%
Advogato	1.94%	0.46%	6.48%	2.54%	3.38%
GnomeML	0.64%	4.66%	6.28%	1.93%	2.78%
Ohloh	0.59%	0.84%	0.89%	0.00%	0.91%
DebianML	6.00%	1.11%	1.27%	0.91%	0.00%

Table 5: Ratio of smushed instances of `foaf:Person` using IFP, relative to the size of the community of the row.

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	5.31%	9.21%	5.47%	3.14%	71.49%
Advogato	4.08%	1.18%	14.20%	2.28%	7.32%
GnomeML	1.74%	10.21%	29.60%	2.55%	7.45%
Ohloh	0.46%	0.75%	1.17%	0.10%	1.65%
DebianML	10.38%	2.40%	3.40%	1.64%	22.90%

Table 6: Ratio of smushed instances of `foaf:Person` with exactly the same `foaf:name`, relative to the size of the community of the row.

Source	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
DebianPkgs	5.31%	10.19%	6.12%	5.64%	76.86%
Advogato	4.51%	1.32%	16.05%	4.01%	8.21%
GnomeML	1.95%	11.54%	30.45%	3.92%	7.97%
Ohloh	0.82%	1.32%	1.80%	0.10%	2.28%
DebianML	11.16%	2.69%	3.64%	2.26%	22.90%

Table 7: Ratio of smushed instances of `foaf:Person` combining both techniques, relative to the size of the community of the row.

	Number of		Presence in				
Name	Name vars.	E-mail acc.	DebianPkgs	Advogato	GnomeML	Ohloh	DebianML
Frederic P.	1	3	●	●	●	●	●
Dan K.	2	3	○	●	●	●	●
Jerome W.	2	1	●	○	●	●	●
Raphael H.	2	3	●	○	●	●	●
Person #01	1	1	●	○	●	●	●
Person #02	1	1	●	○	●	●	●
Person #03	2	4	●	○	●	●	●
Julien D.	1	4	●	●	○	●	●
Rob B.	1	2	●	●	○	●	●
Daniel R.	2	1	●	●	○	●	●
Gürkan S.	5	2	●	●	○	●	●
Ricardo M.	2	3	●	●	○	●	●
Ray D.	3	2	●	●	○	●	●
Person #04	1	3	●	●	○	●	●
Person #05	1	3	●	●	○	●	●
Person #06	2	5	●	●	○	●	●
Person #07	1	3	●	●	○	●	●
Person #08	2	2	●	●	○	●	●
Person #09	1	2	●	●	○	●	●
Person #10	2	3	●	●	○	●	●
Federico Di G.	1	2	●	●	●	○	●
Ross B.	1	2	●	●	●	○	●
Person #11	1	5	●	●	●	○	●
Person #12	1	2	●	●	●	○	●
Person #13	1	3	●	●	●	○	●
Person #14	1	2	●	●	●	○	●
Person #15	1	2	●	●	●	○	●
Person #16	1	2	●	●	●	○	●
Person #17	1	2	●	●	●	○	●
Person #18	1	1	●	●	●	○	●
Person #19	1	3	●	●	●	○	●
Francis T.	2	3	●	●	●	●	○

Table 8: Details of the top people by the number of communities they are present in. A filled dot (●) denotes presence in the community. In order to protect privacy, we only print real names of people who have given us explicit permission to do so.

second case, we rely on the smushing process to merge the identities of these users. The number of smushed instances in Table 2 evidence the fact that people post messages to different mailing lists using the same e-mail address.

Although they must be handled with extreme care due to the issues aforementioned, the combined results of the two smushing processes are consistent with the expected ones. For instance, there is a very high overlap between the Debian developers (maintainers of Debian packages) and the Debian mailing lists. Obviously, Debian developers are a relatively small group at the core of the Debian community, thus they are very active in its mailing lists. Another example is the overlap between Advogato and GNOME mailing lists. Advogato is a reputation-based social web site that blossomed at the same time that the GNOME project was gaining momentum. Advogato was passionately embraced by the GNOME developers, who used Advogato to rate each others' development abilities.

We also studied whether there are some people that are present in many of the communities at the same time. We chose communities which are closely related to each other, consequently, we expected a high number of cross-community subscribers. Table 1b evidences that there are several people who are present in many communities. From Table 8 we conclude that almost all the most active open source developers in our dataset are core members of the Debian community. Another interesting fact is that only a few people among the top members of the communities consistently use a single e-mail address and just one variant of their names. This fact proves the difficulty of the smushing process, but also its usefulness.

7 Conclusions and Future Work

In this paper, we explored smushing techniques to spot redundant RDF instances in large datasets. We have tested these techniques with more than 36,000 instances of `foaf:Person` in a dataset automatically extracted from different online open source communities. We have used only public data sources, consequently, these instances lack detailed personal information.

Comparing the figures in Tables 5 and 6, it is clear that the label-based smusher draws more conclusions than the IFP-based one. The number of redundant resources detected by the former is almost always higher than the one detected by the latter. Moreover, when compared to the aggregated figures in Table 7, we observe that the conclusions of the IFP-based smusher are largely contained in the conclusions of the label-based one. This fact can be explained because users with the same e-mail address often happen to have the same name. The difference between figures in Table 6 and 5 are explained by two facts: (a) there are people who have more than one e-mail account, and (b) there are different people with the same name (namesake). Unfortunately, it is not clear which is the influence of each factor. This is an issue, as smushing conclusions derived from (b) are obviously incorrect.

We are aware of the extreme simplicity of our experimentation using label comparison. In our opinion, however, it contributes to show the potential of this smushing technique. We note that it is possible to have more usages for it, for instance, smushing not just by people's names, but also by their publications, their organisations, etc. Surprisingly, the named-based smushing finds a high number of redundant resources even if the comparison strategy for labels (names) is very simplistic (in this case, case-sensitive string equality comparison). More intelligent comparison functions should lead to a higher recall. In this direction, we are evaluating some normalisation functions for names. We have also evaluated classical IR comparison functions that take into account the similarity of the strings (e.g., Levenshtein); nevertheless, their applicability to compare people's names is open to discussion. In general, a smusher algorithm has a natural maximum complexity of $O(n^2)$ due to the need to compare every possible pair of resources. This complexity raises some doubts about their applicability for very large dataset. Generalisation of these techniques to a web-scale will require to find ways to cut down the complexity.

We believe that the ratio of smushing can be further improved if the dataset is enriched with more detailed descriptions about people. Experiments are being carried out to retrieve additional RDF data from semantic web search engines such as SWSE [15] and Sindice [21] as a previous step to smushing. However, this work is still ongoing and we expect to present it in an upcoming publication. We aim to repeat our experiments in other communities apart from the open source one, for instance the Semantic Web community. The ExpertFinder Corpus [14] and the Semantic Web Conference Corpus[7] can be used for this purpose.

We intend to use our smusher to further investigate the potential for optimisations of the smushing process. The way in which these techniques are implemented is critical to achieve a promising performance of the smushing process, specially for very large datasets. In parallel, increasing the precision of smushing will require to study how to enable different smushing strategies to interrelate and reciprocally collaborate. We have started contacts with people from Table 8 asking them to confirm the communities they participate in; we will use their feedback to to measure the recall and the precision of the smushing process.

We acknowledge that any work on data mining, and in particular, identity smushing, raises some important privacy issues and ethical questions, even when the data used is publicly available on the Web. Actually we got very negative feedback from one the top members of open source communities, tagging this research topic as "*immoral*". Obviously we do not share this point of view, but we understand the privacy issues behind this opinion and we have tried to be extremely careful with the personal information that we manage and print.

References

1. D. Berrueta, S. Fernández, and L. Shi. Bootstrapping the Semantic Web of Social Online Communities. In *Proceedings of workshop on Social Web Search and Mining*

[7] `http://data.semanticweb.org/`

(SWSM2008), co-located with WWW2008, Beijing, China, April 2008.
2. J. Breslin, S. Decker, A. Harth, and U. Bojars. SIOC: an approach to connect web-based communities. *International Journal of Web Based Communities*, 2006.
3. D. Brickley and L. Miller. FOAF: Friend-of-a-Friend. `http://xmlns.com/foaf/0.1/`, November 2007.
4. W. W. Cohen, P. Ravikumar, and S. Fienberg. A Comparison of String Distance Metrics for Name-Matching Tasks. In *Proceedings of the workshop on Information Integration on the Web*, pages 73–78, 2003.
5. L. Ding, T. Finin, and A. Joshi. Analyzing social networks on the semantic web. *IEEE Intelligent Systems*, 9, 2005.
6. L. Ding, L. Zhou, T. Finin, and A. Joshi. How the semantic web is being used: An analysis of FOAF documents. In *38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, Track 4. IEEE Computer Society, 2005.
7. D. Eastlake and P. Jones. RFC 3174: US Secure Hash Algorithm 1 (SHA1). Technical report, IETF, 2001.
8. S. Fernández. RDFohloh, a RDF wrapper of Ohloh. In *1st workshop on Social Data on the Web (SDoW2008), co-located with ISWC2008*, Karlsruhe, Germany, October 2008.
9. S. Fernández, D. Berrueta, and J. E. Labra. Mailing lists meet the Semantic Web. In *Proceedings of 1st workshop on Social Aspects of the Web (SAW2007), co-located with BIS2007*, Poznan, Poland, April 2007.
10. L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer Mediated Communication*, 3, 1997.
11. J. Golbeck and M. Rothstein. Linking Social Networks on the Web with FOAF. In *Proceedings of the 17th international conference on World Wide Web (WWW2008)*, 2008.
12. G. A. Grimnes, P. Edwards, and A. Preece. Learning meta-descriptions of the FOAF network. In *3rd International Semantic Web Conference (ISWC2004)*, 2004.
13. M. Hausenblas, W. Halb, and Y. Raimond. Scripting User Contributed Interlinking. In *Proceedings of the 4th workshop on Scripting for the Semantic Web (SFSW2008), co-located with ESWC2008*, Tenerife, Spain, June 2008.
14. A. Hogan and A. Harth. The ExpertFinder Corpus 2007 for the Benchmarking and Development of Expert-Finding Systems. In *Proceedings of 1st International ExpertFinder Workshop*, 2007.
15. A. Hogan, A. Harth, J. Umrich, and S. Decker. Towards a Scalable Search and Query Engine for the Web. In *Proceedings of the 16th international conference on World Wide Web (WWW2007)*, pages 1301–1302, New York, NY, USA, 2007.
16. C. Kiefer. Imprecise SPARQL: Towards a Unified Framework for Similarity-Based Semantic Web Tasks. In *2nd Knowledge Web PhD Symposium (KWEPSY) co-located with the 4th European Semantic Web Conference (ESWC2007)*, 2007.
17. A. Lait and B. Randell. An Assessment of Name Matching Algorithms. Technical report, Dept. of Computing Science, University of Newcastle upon Tyne, 1996.
18. P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language: Semantics and Abstract Syntax. Recommendation, W3C, February 2004.
19. A. Polleres. From SPARQL to rules (and back). In *Proceedings of the 16th World Wide Web Conference (WWW2007)*, pages 787–796, Banff, Canada, May 2007.
20. J. Preece. *Designing Usability and Supporting Sociability.* John Wiley & Sons, Inc, 2000.
21. G. Tummarello, R. Delbru, and E. Oren. Sindice.com: Weaving the Open Linked Data. In *Proceedings of the International Semantic Web Conference 2007 (ISWC2007)*, volume 4825/2008, pages 552–565, Busan, Korea, November 2007.