SIOC in Action
Representing the Dynamics of Online Communities*

Pierre Antoine Champin
LIRIS, Université de Lyon, CNRS, UMR5205,
Université Claude Bernard Lyon 1, F-69622,
Villeurbanne, France
pchampin@liris.cnrs.fr

Alexandre Passant
Digital Enterprise Research Institute,
National University of Ireland,
Galway, Ireland
alexandre.passant@deri.org

ABSTRACT

SIOC — Semantically-Interlinked Online Communities — provides the Semantic Web with a vocabulary for representing activities and contributions of online communities. However, it focuses on the state of online communities at a given time, while a number of Web application put a strong emphasis on the *dynamics* of their components and users, including microblogging, status and geolocation notification, etc. and as also testified by recent efforts on modelling activity streams. This work proposes a new module for the SIOC vocabulary designed to represent the dynamics of actions within online communities. Hence, we provide an *action-centric* view of online communities, while previous work focused on a *document-centric* or *user-centric* one. Furthermore, we align our work with related vocabularies and Web technologies — both in use and emerging — inside and outside the field of Semantic Web technologies.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*semantic web, web 2.0*

General Terms

Standardization

Keywords

semantic web, social web, activity, action, linked data, sioc, streams, traces

1. INTRODUCTION AND MOTIVATION

The Linked Data initiative [5] aims at providing machine-readable *data* on the Web, complementing the human-readable *documents* that constitutes the major part of the current Web. Among others, its goal is to allow computer-based agents to be much more efficient in assisting users and performing automated tasks on their behalf, thus realising the vision of the Semantic Web [4].

Although the amount of Linked Data has been growing quite impressively in the past few years[1], most of it is essentially *static* by nature. We do not mean that this data is not evolving, but that it describes the current state of their domain of interest at a particular instant of time. This contrasts with the importance of time and *dynamics* witnessed recently in popular Web applications.

Indeed, the advent of blogs in the late nineties has strongly shaped the Web by putting the temporal dimension forward, in primarily organising their content along an antechronological timeline. Wikis also give a high importance to time, by keeping track of the modifications made on a page, and allowing readers to see the history of those changes from and to any version. More recently, new trends in Social Web have confirmed this interest in dynamics by providing users with a variety of tools for publishing information about their activity, and retrieving it from their social network. This is especially well illustrated by the initial tag line of the now leading micro-blogging service Twitter[2]: "What are you doing?" as well as by new geolocation information sharing such as foursquare[3] or Gowalla[4].

In order to enable semantic description of the content generated within these services, an important step in bridging the gap between the Semantic Web and the Social Web is the SIOC ontology — Semantically-Interconnected Online Communities [8]. It allows to expose various social Web applications (blogs, wikis, forums, etc.) on the Web of Linked Data, focusing on representing the activities of these communities and the content generated within. It is implemented in a number of popular software and frameworks (see `http://sioc-project.org/`), and combined with FOAF — Friend Of A Friend [9] — this provide a way to represent both social networks and user-generated content using Semantic Web technologies and Linked Data principles.

In this paper, we introduce SIOC-actions, or sioca for short. It is a module for SIOC designed to represent how users of an online community are manipulating the various

---

*The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Líon 2).

I-SEMANTICS 2010, September 1-3, 2010 Graz, Austria

[1] See `http://linkeddata.org/` for an up-to-date account of the linked-data cloud.

[2] `http://twitter.com/`

[3] `http://foursquare.com/`

[4] `http://gowalla.com/`

digital artifacts that constitute the application supporting that community, from blog posts to status updates. We believe that this kind of information can prove valuable in a large range of applications; for example, we have previously discussed the interest of *interaction traces* in fostering group reflexivity and group awareness in collaborative systems [11], providing a base for digital object memories [20], or supporting flexible automated reasoning [12].

The structure of the paper is the following. In section 2, we describe the sioca rationale and vocabulary. In section 3 we describe different scenarios making use of sioca, then in section 4 we discuss related work and compare sioca with these other initiatives. Finally, we conclude and provide some directions for further work.

2. THE SIOC-ACTIONS MODULE

So far, SIOC and its different modules provide mainly a *document-centric* view of online communities. As can be seen in the left column of Figure 1 from the SIOC specification, 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 abstracts concepts are generic enough to represent a great variety of Web applications, but are best understood when instantiated 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 every Forum contains a set of Posts. Every Post (and actually every Item, Container or Space) can be associated with Tag(s) or Category/ies representing their topic. Moreover, we can see in that schema that SIOC represents the UserAccounts[5] 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. This distinction have its importance in the definition of the sioca vocabulary.

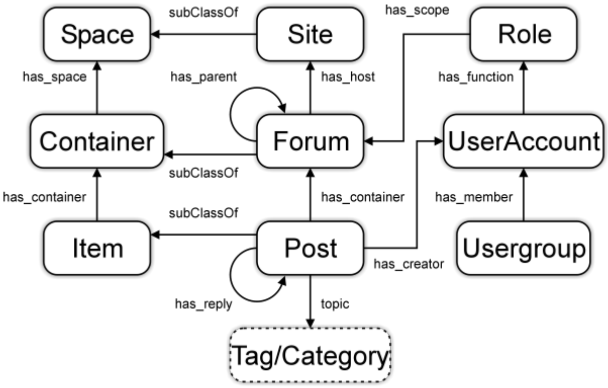

Figure 1: **Main classes and properties of the SIOC vocabulary**

However, the current structure of SIOC cannot capture the related actions associated with the documents. While one can use Dublin Core to state the creation of modification date of an Item (using dc:created or dc:modified), it still focuses on an document-centric (or item-centric) view, rather than on an action-centric one, where the actions that happen in order to create, edit, and share these items would be at the core of the modelling process, as we describe in this paper with the sioca module.

2.1 Actions and their Attributes

The central notion of the sioca module is the Action: a timestamped event involving one user and a number of digital artifacts. It is a subclass of the class event:Event from the the Event Ontology [19].

One point worth noting is that we consider actions to be instantaneous. Although we acknowledge the fact that some actions may actually take some time to be performed, we are only interested in the moment they are taking effect in the considered application — that is, practically, the time the action is registered in the system (*e.g.* the time when an wiki page is saved, and not the time it took to edit it). The rationale of this simplifying assumption is the following:

- First, all applications record the time when an action takes effect, while only some of them record how long it took to perform the action.

- Second, instantaneous actions are expected to be easier to manage in applications using our vocabulary.

We are not defining specific terms for expressing the time and agent of an action; we rather tap existing terms from related ontologies and vocabularies, such as the aforementioned Event Ontology. However, since they do not provide a single uniform way of representing that information, we discuss in the following how to reconcile them in the context of SIOC actions, which are illustrated in Figure 2.

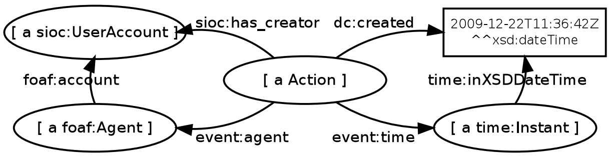

Figure 2: **Two representations of the actor and timestamp of an action**

Per the event ontology, an event can be linked to its actors by the event:agent property, whose range is foaf:Agent. Though we are interested in this information, many application do not provide it, but only the *online account* used to perform the action. In the SIOC ontology, this notion is captured by the sioc:UserAccount class, which may be related to the corresponding agent with the foaf:account property[6]. We therefore require that each action be related to exactly one account through the sioc:has_creator property. We also define an axiom, using the property chain construct from OWL 2 [14], stating that the holder of the account is an event:agent of the action, as illustrated in Table 1 (we use the more concise functional-style syntax of OWL 2).

We have a similar discrepancy for representing the timestamp of an action (if at a structural rather than semantic level). In SIOC, the creation time of an item is usually represented by a literal and linked with property dc:created.

---

[5] Previously sioc:User.

[6] Previously foaf:holdsAccount

```
EquivalentClasses (
    :Action
    ObjectSomeValuesFrom( :_isAction owl:
        Thing )
)
# that artificial property :_isAction
# allows to restrict the following
# property chain to instances of class
# :Action

SubObjectPropertyOf (
    ObjectPropertyChain (
        :_isAction
        ObjectInverseOf (:_isAction)
        sioc:has_creator
        ObjectInverseOf (foaf:account)
    )
    event:agent
)

# the following is in OWL 2 Full
# since dc:created is in fact a
# datatype property

SubObjectPropertyOf (
    ObjectPropertyChain (
        :_isAction
        ObjectInverseOf (:_isAction)
        event:time
        time:inXSDDateTime
    )
    dc:created
)
```

Table 1: OWL 2 axioms for SIOC-actions

On the other hand, the event ontology uses the event:time property to link to an *instance* representing either a time interval or an instant. The latter is overly complicated for our scenario, as we consider only instantaneous actions; furthermore, it can in principle be rebuilt from the information provided by dc:created.

However, this inference can only be represented in OWL 2 Full (since property chains are not meant to be used with *datatype* properties) with other expressive languages, such as RIF [7] or SWRL [15]. It is therefore expected that implementers will use either one, the other or both representations, depending on the level of interoperability they aim to achieve with other vocabularies (see examples in Section 3.4).

2.2 The Objects of Actions

The second important notion of our module is the one of DigitalArtifact, representing the objects manipulated through Actions. Its intent is to represent any component of the Web-based applications targeted by SIOC. It is therefore a superclass of most SIOC classes, such as sioc:Item and sioc:Space, but also sioc:UserAccount[7]. However, for the sake of openness, we do not want to restrict DigitalArtifact to those classes, as they may not cover other kinds of digital artifacts that may emerge in the future. It is not clear, for example, whether Facebook *applications* actually fit one of those classes; however, Facebook applications can obviously be *used*, an action that we should be able to represent with out vocabulary.

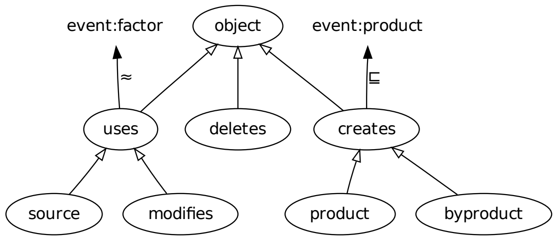

Figure 3: The hierarchy of object properties

The digital artifacts manipulated by an action are called the *objects* of the action, as opposed to the *subject*, which is the actor performing the action. How an action relates to its objects is represented by a hierarchy of properties, rooted in the object property and described by figure 3. Note that an action may be related to several objects, with different subproperties of object for each of them.

The first level of the hierarchy distinguishes three subproperties of object depending on the existence of the object before and after the action. Namely, creates implies that the object didn't exist before the action; uses, that it existed before and after the action; deletes, that it ceased to exist after the action. Of course, creates and deletes both imply that the action is a *cause* of this change in the existence of the object.

Property creates is a subproperty of event:product, which relates an event to "something produced during the event", as per the Event Ontology specification. Note that the properties are not equivalent because object (and hence creates) is restricted to digital artifacts, while event:product can point to any entity (a sound, an abstract situation, etc.). We also distinguish two subproperties of creates: product and byproduct. The former implies that the creation of the object was the function or intent of the action. The latter, on the other hand, indicates that the creation of the object was only a secondary, or even unintended, effect of the action. For example, posting an image on a wiki will create the image resource (the product), but may also create an ancillary wiki-page describing the image (a byproduct). Depending on the context, this distinction may be hard to make or debatable; in those cases, implementers can still fall back to simply using creates.

Property uses relates an action to any digital artifact that is involved in an action without being created or deleted by it. It is related to event:factor, but has no obvious specialisation relation with that property: on the one hand, it is more specific as it is restricted to digital artifacts. On the other hand, it may be more general since event:factors are defined to be "passive factors" of an event. However, we do not exclude that some digital artifacts may have an active role in some actions (for example, bots on a IRC channel may react to a command issued by the user).

We also provide two useful subproperties of uses. Property source is typically associated with a creates property; it means that the source object has served as a primary source

---

[7] Recall that sioc:UserAccount do not represent persons, but rather their account in the application, which is indeed a digital artifact.

of content for the created object. It allows to describe common use patterns on blogging ("reference", "traceback") and micro-blogging ("retweet") applications. Property modifies means that the resource was not only involved in the action, but also that it was significantly altered by it. What "significantly" really means depends of course on each application; for example, changing the license or visibility of an image on a photo-sharing application may not be considered as a significant change of the image (even if the underlying data structure has indeed been modified), while changing the title of the image could be considered as being significant.

3. SIOC-ACTIONS IN USE

In this section we present how SIOC actions can be used to model different existing applications, and what benefit this brings to them.

3.1 Wikis

An important feature of some wikis such as Wikipedia[8] is that they keep track of all modifications made on their pages. While SIOC can be used to represent the current state of a wiki, sioca is therefore relevant for representing the history of each page, in addition to the use of existing properties to link a page to its previous versions [16]. As a proof of concept, we developed a prototype extracting the history of any Wikipedia page, and returning an RDF representation using the sioca vocabulary. This prototype can be tested at `http://champin.net/wsgi/siocat/`.

This export of Wikipedia page histories considers two kinds of digital artifacts: pages and revisions. Pages are mutable objects, whose content changes over time, and which are identified by the usual Wikipedia URI (*e.g.* `http://en.wikipedia.org/wiki/Lyon`). Revisions are immutable objects, corresponding to one particular version of a page; they are identified by revision-specific URIs (*e.g.* `http://en.wikipedia.org/wiki/Lyon&oldid=332929880`). Unlike pages, revisions are never modified once created, they are merely deprecated by newer revisions.

Each action is therefore linked to the page (using the modifies property) and to the corresponding revision (using the creates property[9]). They are also linked to the timestamp of the action, as well as the Wikipedia account responsible for the modification. In the future, axioms could be added to link these actions to the existing modelling of previous and next pages in wikis using SIOC, entailing relationships between the *action-centric* and the *item-centric* principles of modelling wiki versioning with SIOC, that we investigated in [16].

3.2 Developer Communities

Software developer communities have a history of focusing on actions. Indeed, Version Control Systems (VCS), the central tools of those communities, are dedicated to keeping track of actions (usually named "commits" or "patches") modifying a source tree. We demonstrate below how our vocabulary allows to capture the essential features of those actions, and hence to describe the content of VCS on the Web of linked data. This is also implemented in the prototype described above.

An important notion in modern VCS is that of *atomic commit*: related actions on multiple files should always be applied together. For example, the renaming of a function in a file is usually reflected in other files invoking that function; the modification must therefore be either applied on all files or not at all. By allowing an action to have several objects, with a different effect (creates, modifies, deletes) on each of them, our vocabulary is well adapted to represent atomic commits.

The notion of revision, as introduced above to represent wikis, is not always relevant in the context of VCS. It is so in centralised systems, such as Subversion[10], where each commit creates a new state of the source tree. Decentralised VCS such as Bazaar[11] or Git[12], on the other hand, allow each user to have their own version of the source tree depending on the patches they imported from various other users. There is no shared identification of states of the source tree. Consistency is guaranteed by maintaining an acyclic graph of *dependencies* between patches, so that importing a patch will require other ones on which it builds.

Although we do not define a specific term for representing dependencies between actions, the openness of RDF makes it quite easy to provide such a term in a separate vocabulary. Furthermore, other relations between VCS actions and other resources (such as tickets in a bug tracking system) could as easily be added, even if those resources were not hosted by the same application.

3.3 Social Websites

A number of social web applications put a growing emphasis on temporal information and actions. From specialised applications like last.fm[13] tracking music listening habits, to general purpose statuses in Twitter or Facebook[14], a number of users' actions are recorded and broadcast to their social network.

Since those actions take place in the digital environment of social Web applications, they invariably involve digital artifacts (contents, comments, statuses), if only to *use* them (such as a music file or a Facebook application). As with developer communities, our vocabulary is a generic base that can be used as is to provide a generic description of actions in social applications, or extended to provide more detail (see Section 4.3).

3.4 Querying SIOC-actions

The first benefit of a common language to represent actions is obviously the increased interoperability of the tools allowing to cope with that language, such as queries, presentation templates, etc. Using RDF(S)/OWL languages allows, in turn, to use standard and generic technologies to build those tools, notably SPARQL [18] for querying actions, as well as Fresnel [17], Tal4Rdf [10] of LESS[15] for presenting them in a user-friendly way.

Another benefit brought by RDF, which we already pointed out regarding VCS, is its inherent openness. It first allows to

---

[8] `http://en.wikipedia.org/`

[9] Whether the revision is a product or a byproduct depends on one's focus, so we do not commit to one or the other interpretation.

[10] `http://subversion.tigris.org/`

[11] `http://bazaar-vcs.org/`

[12] `http://git-scm.com/`

[13] `http://lastfm.com/`

[14] `http://facebook.com`

[15] `http://less.aksw.org/browse`

mix different vocabularies in order to capture several aspects of the same resource. Second, it allows to link resources from different applications in order to elicit relations that are at best implicit in each separate application [5]. Not only can information be exported outside the boundaries of information silos, but links across sources augments the value of each source by virtue of the network effect.

```
DESCRIBE ?action
WHERE {
  <http://champin.net/foaf.rdf#pa>
    foaf:account ?user .
  ?action event:time ?instant ;
    sioc:has_creator ?user .
  ?instant time:inXSDDateTime ?when .
  FILTER (?when > "2009-11-30"^^xsd:date)
}
```

Table 2: Querying all recent actions by Pierre-Antoine Champin.

```
SELECT ?who ?what ?when
WHERE {
  ?action dc:created ?when ;
    sioc:has_creator ?user ;
    ?what <http://apassant.net/blog
        /2009/11/26/decoding-short-urls> .
  ?what rdfs:subPropertyOf :object .
  ?agent foaf:account ?user ;
    foaf:name ?who .
}
ORDER BY ?when
```

Table 3: Querying all actions on a blog post.

```
DESCRIBE ?action
WHERE {
  OPTIONAL {
    ?action dc:created ?when .
  }
  OPTIONAL {
    ?action event:time ?instant.
    ?instant time:inXSDDateTime ?when .
  }
  FILTER (
    bound(?when)
      && ?when > "2009-12-24"^^xsd:date
      && ?when < "2009-12-26"^^xsd:date
  )
}
```

Table 4: Querying all actions performed on Christmas day.

Table 2 to 6 provide some example queries that could be used, provided a repository using the sioca vocabulary[16]. The query in Table 2 retrieves recent actions involving a given person, regardless of the the various applications this person may have used – as long as we can relate his different online accounts to the URI identifying him as a person.

```
SELECT ?who
WHERE {
  ?action1 :modifies <http://fr.wikipedia.
      org/wiki/Lyon> ;
    sioc:has_creator :user1 .
  ?action2 :modifies <http://en.wikipedia.
      org/wiki/Lyon> ;
    sioc:has_creator :user2 .
  ?who foaf:account ?user1, ?user2 .
}
```

Table 5: Querying all persons having modified both the french and English versions of the Wikipedia page of Lyon.

```
SELECT ?who
WHERE {
  ?who foaf:account ?user .
  ?action :modifies ?page ;
    sioc:has_creator :user .
  ?paper a swrc:InProceedings ;
    dc:creator ?who ;
    foaf:topic ?topic .
  ?topic foaf:page ?page .
}
```

Table 6: Querying all persons having written about a topic and modified that topic in Wikipedia.

The query in Table 3 retrieves all the actions performed on a given blog post. Those actions may include the creation and updates of this post by its author, comments made by readers on the same blog. But it may as well include comments made in other blogs and quoting that post, the creation of shared bookmarks pointing to it, etc. The only limit is the number of action sources available to the query engine.

Note that those queries assume a fixed representation of time: using the event ontology for Table 2, or the Dublin Core vocabulary as suggested by SIOC for Table 3. The query in Table 4 gives another example retrieving all actions performed on Christmas day, regardless of their representation of time.

Those examples only scratch the surface of how SIOC actions can be used to answer complex queries on a collection of actions spanning across application. One could query for all persons having modified two given wiki pages, as illustrated by Table 5. Linking to other data from the linked data cloud, one could look for people having written a scientific paper related to a given topic, and having also altered the Wikipedia page of this topic, as shown in Table 6.

Another example in the field of VCS could be: finding all the persons having solved a bug that they had raised themselves. In the field of social networks, one could ask what kind of music has been recently listened by their friends, looking for “friendship” relations in different social applications. Finally, one could imagine to ask what kind of music is most frequently listened by people fixing bugs in a particular software...

---

[16] Prefixes have been omitted in the queries.

4. RELATED WORK

4.1 Representation of Events

We have chosen to base our vocabulary on the event ontology [19]. This ontology is not by far the only one available to represent events in RDF, as pointed out by Shaw *et al.* [21]. The authors of that paper give a nice overview of different existing proposals, their different ontological commitments, and provide a pivot vocabulary: the LODE ontology.

The reasons why we kept the event ontology as a reference for our own vocabulary are the following. First, the comparison proposed by Shaw *et al.* confirmed that its ontological commitment and level of detail fitted our needs better than other proposals (except for LODE itself, see below). Second, this vocabulary is still widely used in the linked data community, and interoperability may be easier to reach with that vocabulary than with LODE.

Finally, all the terms, but one, that we are borrowing from the event ontology, are semantically *equivalent* to a term from the LODE ontology. Should LODE gain popularity over the event ontology, the migration to that new vocabulary would therefore be straightforward. The only exception is lode:involved, which has no direct correspondence to sioc:factor, and contrarily to the latter, can be considered as a super-property of object. We have therefore included this axiom in our ontology.

4.2 SPARQL and SIOCa

The examples from the previous section raise a number of problems that are outside the scope of this paper, but have been to some extent addressed elsewhere.

First, it may seem from our example queries that they require all actions to be stored in a single repository, which is obviously unrealistic if we aim at targeting a great number of Web-scale applications[17]. However, several proposals [6, 13] have been made to execute SPARQL queries over the whole Web of linked data by discovering relevant sources opportunistically, while the SPARQL Working Group has recently proposed an extension for federated queries[18].

Another problem with using SPARQL to query SIOC actions is that SPARQL query language is usually not considered as a good fit to querying temporal data. Indeed, built-in operators on temporal datatypes only allow for comparison; computing the duration between two actions is not feasible in standard SPARQL. Furthermore, even comparing events temporally can become quite complex, as illustrated in Figure 7. Several proposals have been made to enhance SPARQL with this temporal aspect [22] or to provide a dedicated query language [3]. However, those efforts consider time not as a part of RDF data, but as an orthogonal dimension, where each RDF triple is only valid during a certain period of time. They are therefore not directly applicable to our scenarios.

4.3 Activity Streams and Traces

Aside from the linked data and semantic Web communities, key players in social Web applications have also seen the benefits in fostering interoperability and exchanging information about actions and activities. This has resulted in a joint effort to define a common format for Activity Streams [2, 1]. The proposed format is based on the popular XML-based Atom format: each activity (equivalent to our notion of Action) is described by an Atom entry extended with specific elements and attributes. This format being based on XML, we argue that it not as prone as sioca to integrate with the growing amount of RDF-based linked data. Although the underlying conceptual model seems[19] quite similar to ours, activities are restricted to involve exactly one object. As illustrated by the examples in section 3, we consider this restriction to be too strong.

```
SELECT ?action1 ?action2 ?action3
WHERE {
  ?action1 dc:created ?when1 .
  ?action2 dc:created ?when2 .
  ?action3 dc:created ?when3 .
  FILTER (?when1 < ?when2 && ?when2 < ?
      when3)
}
```

Table 7: A SPARQL WHERE clause stating that action2 must happen between action1 and action3.

On the other hand, Activity Streams define a number of interesting notions that are not currently present in our vocabulary. A *stream* is a collection of activities, providing general information about the activities it contains. In particular, a stream has a *subject*, which can be either a digital artifact or an agent, and is the common feature between all the activities of the stream. Activities may also have a *target*, *e.g.* the album to which a photo is posted. This, as well as an extensive list of *verbs* defined in [1], are good candidates for defining subproperties of object.

5. CONCLUSION AND FURTHER WORK

In this paper, we have proposed a vocabulary to represent the dynamics of online communities, by representing the actions performed by the users of those communities. This vocabulary complements the SIOC ontology, a popular vocabulary for representing the static features of online communities, and provides him with an *action-centric* view of online communities, where it used to focus mainly on an *item-centric* one. We illustrated how actions can be modelled and queried in different kinds of online communities, and provided a link to a running prototype demonstrating the extraction of SIOC-action linked data from heterogeneous sources, including activity streams.

As discussed in section 3, further refinement of our vocabulary may be required by different applications. The Activity Streams [1] effort provides a base for such extensions in the domain of social applications. We are considering to propose an extension of both SIOC and sioca dedicated to developer communities. This will relate to other existing vocabularies, such as DOAP[20]. Moreover, as Activity Streams are rapidly gaining momentum in the domain of social Web applications, we are also planning to provide translation services between both formats. As stated in Section 4.3, there is indeed a large overlap between them.

---

[17] This remark is not specific to sioca, but to Semantic Web applications dealing with distributed data sources in general

[18] `http://www.w3.org/TR/2010/WD-sparql11-federated-query-20100601/`

[19] No conceptual model is explicitly described; it can only be inferred from the proposed syntax.

[20] `https://trac.usefulinc.com/doap`

Finally, we will continue to explore innovative applications consuming actions in the form of interaction traces, as the ones proposed in [11, 20, 12]. From that perspective, not only would SIOC-actions add a new dimension to the Web of linked data, but their relations to existing Linked Data would open new perspectives to trace-based applications.

6. REFERENCES

[1] M. Atkins, D. Recordon, C. Messina, M. Keller, A. Steinberg, and R. Dolin. Atom activity base schema (draft). Internet-Draft, Dec. 2009. `http://martin.atkins.me.uk/specs/activitystreams/activityschema`.

[2] M. Atkins, D. Recordon, C. Messina, M. Keller, A. Steinberg, and R. Dolin. Atom activity extensions (Draft). Internet-Draft, Sept. 2009. `http://martin.atkins.me.uk/specs/activitystreams/atomactivity`.

[3] E. Baratis, E. G. M. Petrakis, S. Batsakis, N. Maris, and N. Papadakis. Toql: Temporal ontology querying language. In N. Mamoulis, T. Seidl, T. B. Pedersen, K. Torp, and I. Assent, editors, *SSTD*, volume 5644 of *Lecture Notes in Computer Science*, pages 338–354. Springer, 2009.

[4] T. Berners-Lee. Semantic web roadmap. Draft, W3C, Sept. 1998.

[5] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data – The Story So Far. *International Journal on Semantic Web & Information Systems*, 5(3):1–22, 2009.

[6] U. Bojars, A. Passant, F. Giasson, and J. G. Breslin. An architecture to discover and query decentralized rdf data. In S. Auer, C. Bizer, T. Heath, and G. A. Grimnes, editors, *SFSW*, volume 248 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[7] H. Boley, G. Hallmark, M. Kifer, A. Paschke, A. Polleres, and D. Reynolds. RIF core dialect. W3C proposed recommendation, W3C, Oct. 2009. `http://www.w3.org/TR/rif-core/`.

[8] J. G. Breslin, A. Harth, U. Bojars, and S. Decker. Towards semantically-interlinked online communities. In A. Gómez-Pérez and J. Euzenat, editors, *ESWC*, volume 3532 of *Lecture Notes in Computer Science*, pages 500–514. Springer, 2005.

[9] D. Brickley and L. Miller. FOAF Vocabulary Specification, January 2010.

[10] P.-A. Champin. T4R: Lightweight presentation for the Semantic Web. In G. A. G. Chris Bizer, Sören Auer, editor, *Scripting for the Semantic Web, workshop at ESWC 2009*, June 2009.

[11] D. Clauzel, K. Sehaba, and Y. Prié. Modelling and visualising traces for reflexivity in synchronous collaborative systems. In *International Conference on Intelligent Networking and Collaborative Systems (INCoS 2009)*, pages 16–23. IEEE Computer Society, Nov. 2009.

[12] A. Cordier, B. Mascret, and A. Mille. Extending Case-Based Reasoning with Traces. In *Grand Challenges for reasoning from experiences, Workshop at IJCAI'09*, July 2009.

[13] O. Hartig, C. Bizer, and J. C. Freytag. Executing SPARQL queries over the web of linked data. In A. Bernstein, D. R. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, editors, *International Semantic Web Conference*, volume 5823 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2009.

[14] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. OWL 2 web ontology language primer. W3C recommendation, W3C, Oct. 2009. `http://www.w3.org/TR/owl2-primer/`.

[15] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: a semantic web rule language combining OWL and RuleML. W3C member submission, W3, May 2004. `http://www.w3.org/Submission/SWRL/`.

[16] F. Orlandi and A. Passant. Enabling cross-wikis integration by extending the sioc ontology. In *SemWiki*, volume 464 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[17] E. Pietriga, C. Bizer, D. Karger, and R. Lee. Fresnel: A browser-independent presentation vocabulary for RDF. In *Lecture Notes in Computer Science*, volume 4273, page 158, Athens, GA, USA, Nov. 2006.

[18] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. W3C recommendation, W3C, 2008. `http://www.w3.org/TR/rdf-sparql-query/`.

[19] Y. Raimond and S. Abdallah. The Event Ontology, Oct. 2007. `http://motools.sf.net/event/event.html`, accessed on 22/12/2009.

[20] L. S. Settouti, Y. Prié, D. Cram, P.-A. Champin, and A. Mille. A trace-based framework for supporting digital object memories. In *1st International Workshop on Digital Object Memories (DOMe'09) in the 5th International Conference on Intelligent Environments (IE 09)*, July 2009.

[21] R. Shaw, R. Troncy, and L. Hardman. LODE: Linking open descriptions of events. In A. Gómez-Pérez, Y. Yu, and Y. Ding, editors, *ASWC*, volume 5926 of *Lecture Notes in Computer Science*, pages 153–167. Springer, 2009.

[22] J. Tappolet and A. Bernstein. Applied temporal rdf: Efficient temporal querying of rdf data with sparql. In L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou, and E. P. B. Simperl, editors, *ESWC*, volume 5554 of *Lecture Notes in Computer Science*, pages 308–322. Springer, 2009.

Appendix: Prefixes used in this paper

dc:	`http://purl.org/dc/terms/`
event:	`http://purl.org/NET/c4dm/event.owl#`
foaf:	`http://xmlns.com/foaf/0.1/`
lode:	`http://linkedevents.org/ontology/`
sioc:	`http://rdfs.org/sioc/ns#`
time:	`http://www.w3.org/2006/time#`
$\emptyset$	`http://rdfs.org/sioc/actions#`