

SIOC BROWSER - TOWARDS A RICHER BLOG BROWSING EXPERIENCE

Uldis Bojars⁺, John G. Breslin⁺ and Alexandre Passant^{*^}

⁺ *DERI, National University of Ireland, Galway, Ireland*

^{*} *EDF, Recherche and Développement, Clamart, France*

[^] *LaLICC, Université Paris IV, Paris, France*

uldis.bojars@deri.org, john.breslin@deri.org, alexandre.passant@edf.fr

1 Introduction to SIOC

1.1 Overview

At present, there are many communities of blogs that are connected to each other in a formal or informal manner, through blogroll links, casual references from posts or via sites set up as aggregators for communities of interest. However, there is much information in such communities that could be formalised for use in searching, browsing and viewing of blog posts either locally or using third-party sites or readers.

The SIOC (Semantically-Interlinked Online Communities, pronounced „shuck“) project¹ (Breslin, 2005) provides methods for interconnecting blogs and blog communities to each other and also to other discussion methods such as forums and mailing lists. It consists of the SIOC ontology², an open-standard machine readable format for expressing the information contained both explicitly and implicitly in internet discussion methods, of SIOC metadata producers for a number of popular blogging platforms and content management systems, and of storage and browsing / searching systems for leveraging this SIOC data.

There are many potential uses for such exporting SIOC data that will be explored. Firstly, SIOC data can be used in blogging publish / subscribe mechanisms, without loss of important associated metadata about authors, comments and related links. Secondly, SIOC can provide a transport mechanism between blogs, or between blogs and other discussion methods such as bulletin boards (since these share many of the same concepts as blogs, but differ in the sense of ownership). Thirdly, SIOC metadata can be detected on blogs using auto-discovery mechanisms, either by metadata crawlers or using browser plugins (a SIOC detector for Firefox will also be highlighted). Fourthly, SIOC can be used to get the most active user or other current states of activity for a set of blogs.

In this paper, we will begin by giving an overview of the SIOC ontology and how it can be used to connect different online community primitives. We will then describe how PHP-based SIOC modules and reusable libraries have been created for a number of popular blogging platforms, including WordPress, Drupal and DotClear. We will then describe the SIOC Browser, and show how this browser can leverage SIOC to provide new ways for visualising posts from blogs and other discussion primitives. Finally, we will present our conclusions and ideas for future work based on this paper.

1.2 The SIOC Ontology

The SIOC ontology has been created using RDF, or the Resource Description Framework³, a metadata format used to create Semantic Web-enabled sites and applications. As such, the SIOC ontology benefits from a number of useful properties of the Semantic Web. The first is that SIOC is an open standard, and can be used by anybody in their internet discussion applications. Another is that the Semantic Web allows the reuse of existing vocabularies, and as a Semantic Web ontology, SIOC can be combined with other popular vocabularies such as the Friend of a Friend (FOAF) ontology⁴ (defining people and what they do) and the Dublin Core (DC) ontology (describing documents and resources).

Using the SIOC ontology, „Sites“ host „Forums“ (or blogs) that contain „Posts“ created by „Users“, and comments or replies can also be expressed as „Posts“. Any of these concepts can be defined as being part of the general concept of „Community“, and therefore SIOC can be used to describe distributed versions of communities, discussion channels and threads across multiple sites.

Two approaches are used to integrate SIOC with other Semantic Web ontologies: (1) SIOC concepts are formally defined as being sub-concepts of other existing external ontology concepts through mappings, so that the properties of these concepts can be reused in SIOC but also so that more semantically-constrained properties can be defined for SIOC alone, and (2) concepts or properties from other ontologies are informally suggested for use in combination with SIOC concepts through our documentation and use case examples. An example of (1) is that the concept `sioc:User` is a sub-type of `foaf:onlineAccount` (i.e. a user account held by a `foaf:Person`), and of (2) is that the properties `dc:title`, `dc:description` and `dc:subject` can be used in conjunction with the other properties of a `sioc:Post`.

In comparison with SIOC, RSS and Atom are commonly-used generic syndication formats whose power lies in their simplicity: any website can be described as a channel, and the content of pages can be described as items. At present, the syndicated content of a blog is retrieved using either RSS or Atom, but these formats can be somewhat limited in that they typically only display the most recent 10-15 posts created on the blog, with links to separate comment feeds only provided by a small number of blogging platforms (e.g. WordPress) - and much semantic information is lost. Blog search engines such as Technorati or aggregators such as Gregarius may therefore only retrieve the latest blog posts when a blog is discovered that has already been existence for some time. Also, the full nature of a blog's structure and content cannot be adequately expressed using RSS.

This is one of the motivations for the use of SIOC to describe all of the past content contained within a blog and its relations to other sites. SIOC is ideally suited to this task as it is more semantically constrained (being designed specifically to describe discussion methods as opposed to any content) and also all historical

content can now be made available. Again, by leveraging the power of the Semantic Web to combine terms, data in SIOC terms can be combined with those already made available via RSS 1.0 and AtomOWL.⁵

1.3 Creating Connections with SIOC

Figure 1 shows some typical internet discussion methods: weblogs, bulletin boards, mailing lists and Usenet groups. These are shown in „clouds“ (e.g. the blogosphere) corresponding to each method. The arrows between the discussion posts may be in the form of internal thread replies, trackbacks or explicit URIs. It should be noted that not only do the discussion systems and protocols differ from one cloud to another, but they can also differ within the clouds themselves (e.g. in the blogosphere, there are many different versions of blogging software, XML-RPC APIs, syndication formats and categorisation being used). At present, there are a number of ways of interconnecting blogs and posts (through blogrolls and trackbacks), but the mechanisms for these are not consistent within the blogosphere and have not been transferred to other discussion methods so far.

SIOC provides a unified vocabulary for content and interaction description: a semantic layer that can co-exist with existing discussion platforms. Using SIOC, various linkages are created between the aforementioned concepts, which allow new methods of accessing this linked data, including:

- *Virtual Forums.* These may be a gathering of posts or threads which are distributed across discussion platforms, for example, where a user has found posts from a number of blogs that can be associated with a particular category of interest, or an agent identifies relevant posts across a certain timeframe.
- *Distributed Conversations.* Trackbacks are commonly used to link blog posts to previous posts on a related topic. By creating links in both directions, not only across blogs but across all types of internet discussions, conversations can be followed regardless of what point or URI fragment a browser enters at.
- *Unified Communities.* Apart from creating a web page with a number of relevant links to the blogs or forums or people involved in a particular community, there

is no standard way to define what makes up an online community (apart from grouping the people who are members of that community using FOAF or OPML). SIOC allows one to simply define what objects are constituent parts of a community, or to say to what community an object belongs (using `sioc:has_part / part_of`): users, groups, forums, blogs, etc.

- *Shared Topics.* Technorati (a search engine for blogs) and BoardTracker (for bulletin boards) have been leveraging the free-text tags that people associate with their posts for some time now. SIOC allows the definition of such tags (using the subject property), but also enables hierarchical or non-hierarchical topic definition of posts using `sioc:topic` when a topic is ambiguous or more information on a topic is required. Combining with other Semantic Web vocabularies, tags and topics can be further described using the SKOS organisation system.⁶
- *One Person, Many User Accounts.* SIOC also aims to help the issue of multiple identities by allowing users to define that they hold other accounts or that their accounts belong to a particular personal identity (via `foaf:holdsOnlineAccount` or `sioc:account_of`). Therefore, all the posts or comments made by a particular person using their various associated user accounts across platforms could be identified.

We will now describe how SIOC data can be produced from blogs and other internet-based discussion methods, leading to the creation of the aforementioned connections.

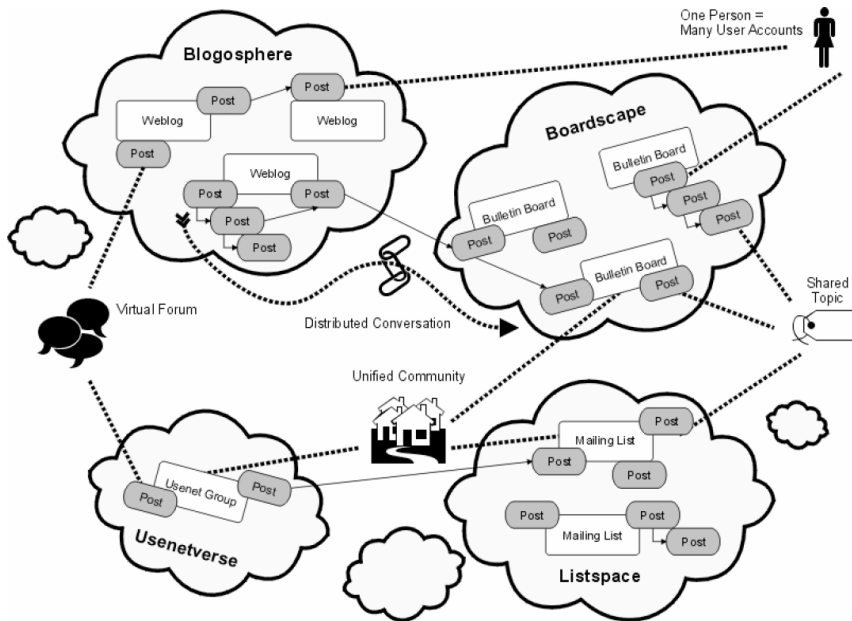


Figure 1: Creating connections between discussion clouds with SIOC

2 How to Create SIOC Data

The power of blogs lies in their number and the large amount of blog data that is available for harvesting. Most blog engines already have RSS export functionality. Since the majority of these blog engines are based on open source software or can be extended by using plugins, it is straightforward to modify existing export functions or create plugins to generate metadata conforming to the SIOC ontology. A SIOC API for PHP (described below) makes the development of such SIOC plugins and exporters as easy as possible.

2.1 SIOC APIs

We aim to overcome a „chicken-and-egg“ problem (no applications without data, and no data without applications) by making it easy to generate and use SIOC data. A SIOC export API for PHP⁷ has been created by the authors. It enables developers to create SIOC export tools without the need to get into technical details about how information is represented in RDF/XML – they are operating at the level of SIOC concepts instead. Thus, developers only have to deal with extracting content from their weblog databases and then passing it to the API that will render SIOC data.

The SIOC PHP API is already being used by SIOC export plugins for the DotClear and b2evolution blog engines. This export API was written in PHP due its popularity as a programming language for web applications, and also because many blog engines are written in PHP. There is a need for SIOC export APIs in other languages, such as Ruby on Rails (as used in the Typo blog engine).

The SIOC API creates and exports SIOC concepts from the blog: authors (sIOC:User plus foaf:Person), posts and comments (sIOC:Post) and the structure of the blog (sIOC:Site plus sIOC:Forum). While a regular weblog contains only a single blog (represented in SIOC as a sIOC:Forum) the Site/Forum architecture allows us to describe the structure of all weblogs and online community sites in a uniform way, e.g., multi-user blog sites such as wordpress.com can be represented as a Site with multiple Forums (blogs).

2.2 Exporters

SIOC plugins export the information about the content and structure of blog sites in a machine readable form. In addition to the plugins mentioned above, SIOC export plugins and tools have been developed for the WordPress blog engine, Drupal CMS, phpBB bulletin board and legacy community formats such as mailing lists.⁸

These plugins or exporters can be developed for any blog engine that you want to export the information from. Plugins can be built on their own or can use the aforementioned SIOC API for PHP. Using the API is the recommended and easiest approach as developers will not have to change the plugin as the SIOC format evolves - all the changes are made once, at the API level. In either case, the exporter or plugin uses the internal logic (functions, classes, etc.) of the blog engine or else direct database access to get information about the weblog, and then uses the SIOC API or custom code to produce SIOC export data.

Most of the exporters implement a SIOC auto-discovery mechanism which allows us to identify web pages that contain SIOC metadata using automated tools such as the Semantic Radar plugin for Firefox.⁹

2.2.1 WordPress

We have created a SIOC export plugin¹⁰ for WordPress - one of the most popular blogging tools. It makes use of existing WordPress PHP functions to access the information from the underlying relational database and generate SIOC metadata in RDF for each concept instance.

The export process is illustrated by example in Figure 2. It shows the export process as implemented in a new version of WordPress SIOC plugin (in development) using a SIOC API for PHP.

In addition to the regular set of SIOC metadata about a blog post, the plugin also mines the post body for hyperlinks and links to additional machine-interpretable information (e.g., those created by semiBlog¹¹). The plugin exports such links via SIOC as `sioc:links_to` and using `rdfs:seeAlso` references in RDF.

WordPress SIOC plugin can also be extended to work with multi-user blogging sites powered by WordPress Multiuser¹², allowing the description of large communities of bloggers.

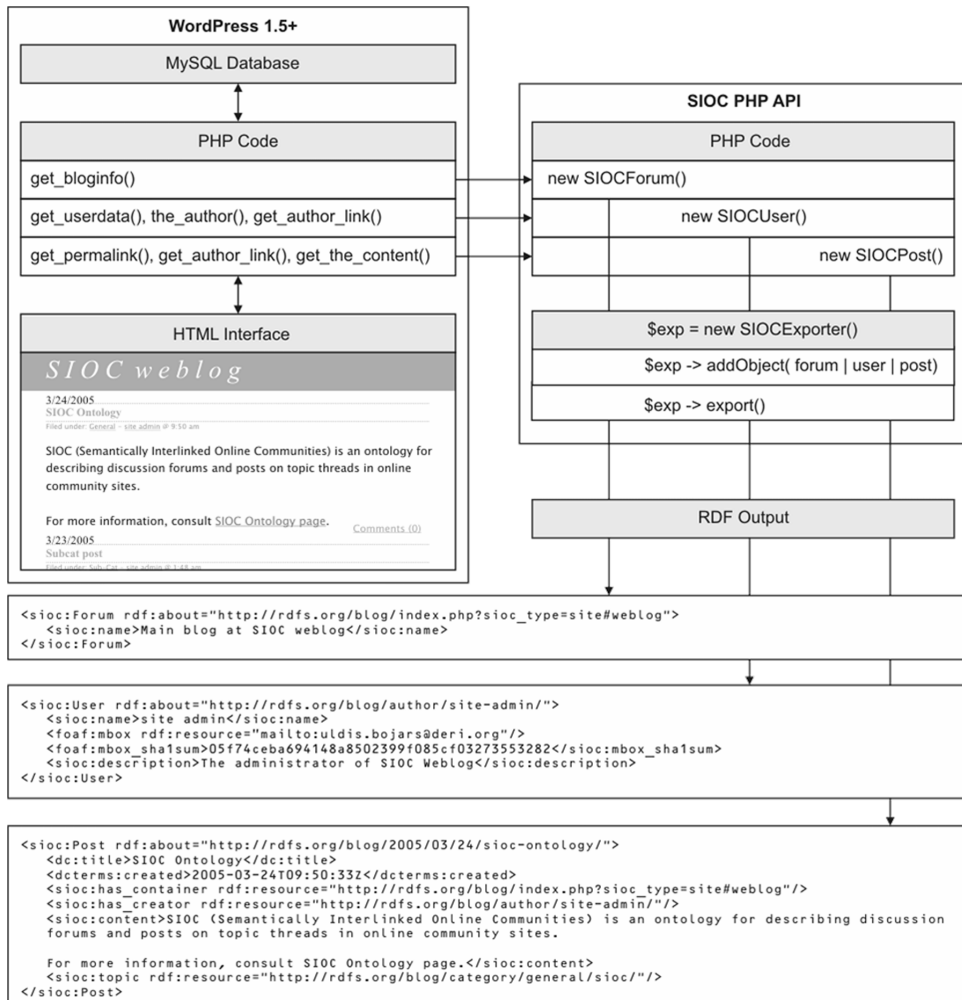


Figure 2: The architecture of the SIOC plugin for WordPress

2.2.2 DotClear

DotClear is another blogging platform, widely used in the French blogosphere and written in PHP. Using the PHP API, a SIOC exporter was written¹³ and packaged as a plugin so that users can install it from their software backend. The exporter creates data for users, posts, comments and trackbacks. It also extracts external links from any blog post using `sioc:links_to` property, and uses both categories or tags (requiring the tag plugin to be installed) to define `sioc:topic(s)`. Finally, it also features autodiscovery links that will point to the SIOC data corresponding to the currently viewed page.

As DotClear is currently being rewritten as a new version with a different plugin architecture, a SIOC exporter for DotClear2 will be created as soon as a stable version is released. In the meantime, this plugin will be packaged as part of a more generic Semantic Web package for DotClear, including FOAF, SIOC and other Semantic Web capabilities.

2.2.3 Drupal

There is also a plugin for Drupal, which can be used to export SIOC data from Drupal blogs and forums.¹⁴ As Drupal can be used as a multi-user blogging platform, the plugin will export all blogs and all users account, so that each post can be clearly identified by its users. This plugin is currently used in anime.ie, a community website with 120 users.

2.2.4 b2evolution

b2evolution is a multi-blog platform that evolved from the same roots as WordPress - from b2/cafeblog. A SIOC exporter for b2evolution¹⁵, built upon the SIOC API for PHP, is now available.

3 How to Use and Browse SIOC Data

SIOC data can be viewed, used and queried in different ways. All SIOC data is written in RDF and should be processed as such. The existing APIs for processing RDF such as RAP (in PHP)¹⁶ or Redland (written in C, with bindings for C#, Java, Objective-C, Perl, PHP, Python and Ruby)¹⁷ can be used to read and process data from SIOC-enabled websites. Thus, we wrote and utilised a number of different browsing and rendering interfaces to explore the expressiveness of SIOC in a user-friendly way.

A community of people that are interested in SIOC and its uses has recently been formed.¹⁸ This site provides more implementations to work with and real-world data that can be explored using SIOC-enabled tools such as the browsers described below. Examples of projects using SIOC are TalkDigger¹⁹ and OpenLink Data Spaces.²⁰

3.1 Semantic Radar

To facilitate end-user access to SIOC data, we have developed the Semantic Radar²¹ - a Firefox browser extension that detects presence of SIOC, FOAF and DOAP data in a webpage, and alerts a user who then has the possibility to browse data in an online SIOC browser. RDF auto-discovery information generated by data export plugins provides a machine-readable indication of the presence of SIOC data.

The radar application can ping the PingTheSemanticWeb website²², an online service that collects, stores and distributes links to RDF documents for every ping, and this is an efficient way to find and index SIOC data over the Web. URLs indexed by the ping service can be re-used by other web services and can also be viewed on its website. SIOC-enabled websites can also be registered on the SIOC community wiki²³ and this wiki page can then be used by Semantic Web tools as a starting point for crawling and exploration.

3.2 Browsers Overview

Since people can quite easily create SIOC data from their weblog using the described exporters or the API introduced in the previous chapter, we will now consider different ways to query and view this data.

SIOC browsers²⁴ allow people to browse and receive additional information from SIOC data sources or data stores. Browsers can work in two modes - on-the-fly mode and crawler mode - or can use a combination of both. The former displays the SIOC data received from a community site (thus providing a uniform interface to all SIOC-enabled sites) while the latter stores SIOC data in a repository, allowing one to make more complicated queries via the use of SPARQL²⁵, the leading Semantic Web query language.

SIOC data can also be displayed in a graphical format – as a timeline of content posts or as a graphical network of relations between sites, posts and topics. These graphical interfaces offer another user-friendly way to view SIOC data and the crawler mode browser may be extended to provide both a text and graphical view of the same information. Moreover, as SIOC offers an open data format using a formal description in RDF, everyone can create their own tools as appropriate to their needs. Thus, the browsers provided here are an example of what can be done with SIOC, and give a nice overview of its different capabilities.

3.3 On-the-Fly Mode

The on-the-fly or live SIOC browser is a simple and effective way to explore community information available in SIOC. It gives a user-friendly look at the internal structure of the data without requiring the viewers to dive into a more complex RDF/XML syntax.

The on-the-fly SIOC browser takes a URL of a SIOC data source, e.g., from a SIOC plugin to the blog engine, and presents viewers with a web page showing a hierarchy of the concepts and links contained within it. For a blog post, this information would include all that is important to know: title, author, date, content, topic

categories and tags, and information about replies (comments and trackbacks) to that post. An sample post as seen in the SIOC browser is shown in Figure 3.



Figure 3: Example post displayed in the SIOC browser

Some of the links in SIOC data, called `rdfs:seeAlso` links, point to additional machine readable information (in RDF and SIOC). A browser can follow these links and display this SIOC information in the same user-friendly way.

3.4 Crawler Mode

3.4.1 Overview

Thanks to the SIOC API and plugins for the various blog platforms that have been introduced already, people can easily create SIOC exports of their websites. Thus, a growing number of sources provide SIOC data on the Web. Instead of browsing each of the SIOC data pages separately as in the on-the-fly browser, the general idea behind this second kind of browser is to collect SIOC data from different sources and put it in a database. Then, one will be able to query this database (e.g., using a SPARQL query language) and then see the information collected from a number of sources (blogs, forums, etc.). This enables a more advanced use of SIOC information, such as discovering relations between blogs, people and topics, or collecting summary information. Such an architecture can be seen as a „Semantic Web Technorati“.

There are different parts involved in this browsing experience (see Figure 4):

- A SIOC crawler, used to fetch data from different SIOC-enabled websites;
- An RDF store in which crawled data will be saved so that we can make queries on it;
- The browser itself which in this architecture consists of:
 - Different query configuration files - the XML files containing metadata about the query plus parts of SPARQL and PHP code used to render and format query results;
 - The browser engine that will interpret these configuration files and run the queries over the RDF store to display results in a user-friendly interface. Queries are sent to the store using AJAX²⁶, and the browser uses a common interface whatever the query is so that we get a uniform look and feel for both textually and graphically rendered queries.

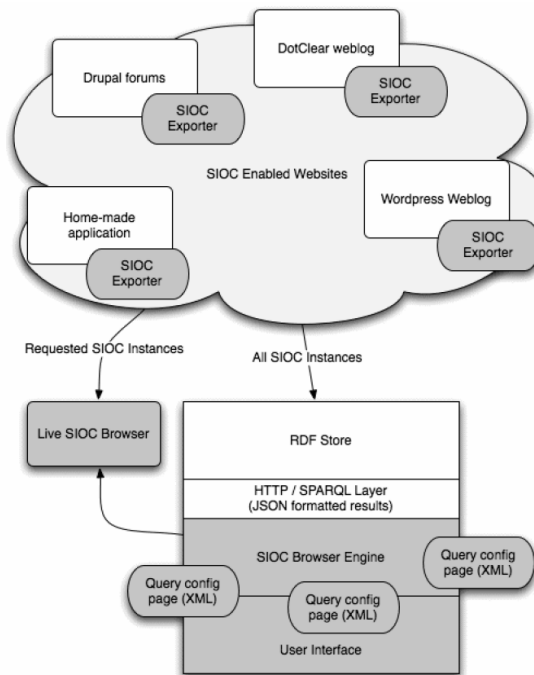


Figure 4: The components of the SIOC browser

3.4.2 Crawling, Storing and Querying SIOC Data

In order to retrieve a weblog SIOC export, a custom RDF crawler for SIOC data²⁷ has been written in Python. The SIOC data is collected by a crawler or spider that traverses the Web and retrieves any SIOC data it finds. The crawler starts with a list of „seed“ SIOC URLs and follows `rdfs:seeAlso` links used to point to more SIOC and RDF data. Since the data pages exported by SIOC plugin have links to each other the crawler will collect all the SIOC data that these sites provide.

This is a generic principle for crawling RDF documents and we could use a generic RDF crawler. But additional knowledge about the structure of SIOC data allows us to create crawlers with advanced functionality, e.g., incremental retrieval of new SIOC data.

Since we deal with RDF data, all content fetched is stored in an RDF store. An RDF store, also known as a triple store, can be seen as a database dedicated to Semantic Web content and it keeps the structure of RDF by storing triples (or statements). What differentiates it from a relational databases approach is that RDF stores are generic - they can store any information expressed in RDF and do not require redesign to accommodate changes to the structure or content of the data to be stored. A lot of stores are currently available, which differ in their performance and features. In our case, the main requirements are support for the SPARQL query language, the SPARQL protocol over HTTP²⁸, and the ability to format SPARQL query results using JSON serialization (Clark, 2006).

These requirements form a nice abstraction level between the browser and the RDF store, and allow us to easily change or adapt an RDF store without changing anything on the browser except the SPARQL endpoint URL. Thus, our first implementation was based on Joseki²⁹, but we also made it run with 3Store³⁰, without any change in the browser code. Since data is present in the store, we can make queries on it. For example, the following SPARQL query applied to our RDF store will retrieve all comments (and related posts) Alexandre Passant wrote since the 1st of August 2006.

```
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT ?post ?reply ?date ?reply_date
WHERE
{
  ?post sioc:has_reply ?reply .
  ?post dcterms:created ?date .
  ?reply sioc:created_at ?reply_date .
  ?reply foaf:maker ?_author .
  ?_author foaf:name ?name .
  FILTER (xsd:dateTime(?date) >= „2006-08-
01T00:00:00Z“^^xsd:dateTime) .
  FILTER REGEX(str(?name), „^Alexandre Passant$“)
}
```


This query shows some advantages of using SIOC and crawled data:

- Firstly, as all weblogs express their data using the same vocabulary, the query pattern could be applied to any blog without distinction;
- Secondly, the query is applied not only to one weblog but to data from many decentralised weblogs that can now be queried together as if they were collected in a single database;
- Thirdly, the query is applied to all posts and comments published on each of these weblogs (compared with RSS where only the last 10-15 posts can be fetched);
- Finally, the comments in SIOC are integrated with the rest of data (compared with RSS where comments are in separate feeds and hard to integrate with post feeds), and we can look at all comments from the same user in all crawled weblogs.

Moreover, the query can involve different ontologies such as FOAF, SIOC and DC - as mentioned in the introduction SIOC has mapped and reused a number of other popular ontologies. Indeed, a `sioc:Post` is identified by both `sioc:has_creator` (the `sioc:User` involved in the post) and `foaf:maker` (the `foaf:Person` that holds this account). The former can identify all the posts by the same user on a weblog while the latter allows us to identify posts by the same person across the Web. The PHP API exports both relations for each `sioc:Post`, while using only `foaf:maker` for comments from unregistered users of the platform as they do not have a user account on the system.

3.4.3 User Interface

In order to avoid users having to learn SPARQL to type their queries and then writing scripts to parse the results, we wrote our browser with a number of pre-defined queries and a user-friendly interface.

The first query simply displays the list of weblogs currently stored in the system. It consists of retrieving all `sioc:Forum` instances from the store, with their associated name using `dc:title`. Then, users will be able to visit the original blogs using this list.

The second one displays a „tag cloud“ of topics. As on Technorati, the goal is to display the most used topics from the blogs we crawled. For each tag, users can see associated posts, and can then browse them using the live browser or by visiting the original blog. Regarding this, at the moment we use the label of the topics to create the tag cloud, as there is not yet any semantic information in weblogs as to what a topic (category or tag) is about. However, we could imagine defining unique identifiers or URIs for topics and describing categories or tags with SKOS³¹, or using - in a closed space - some ontologies to link a `sioc:topic` to its concepts (Passant, 2006), so that we will obtain a unification of topics amongst blogs.

Another textual query we wrote aims to order posts by the number of comments they have within a defined timeline (using a SPARQL `FILTER` parameter). Thus, we can see which posts were the most 'popular' during a given period - a nice way to see the 'hot topics' of the moment.

To enhance the visualisation interface, we also wrote a set of queries that are rendered graphically. The first one demonstrates some basic statistics about topics and their distribution in our triple store. It uses the same SPARQL query as the tag cloud, but the PHP parsing is different and limits the result to the first 10 popular topics. Then, results are rendered to a dynamically flash chart, using the PHP/SWF Charts API.³² Once again, this is an interesting (and more friendly) way to see some of the advantages of the crawling mode browser, as we can instantaneously get statistics about data fetched from decentralised weblogs.

Query

```

PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT DISTINCT ?post ?post_topic ?topic_label
WHERE {
    ?post sioc:topic ?post_topic .
    ?post_topic rdfs:label ?topic_label .
    ?post dcterms:created ?post_date
}

```

Results

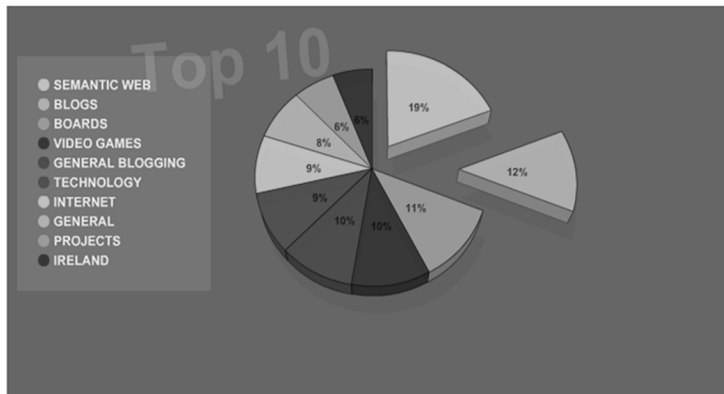


Figure 5: Topics results from crawled SIOC data

Finally, the latest query interface in this browser uses the community aspect of SIOC and the notion of posts and comments to display relations between people among a set of blogs. As SIOC offers the ability to represent posts, comments and trackbacks from any blog in a semantically well-defined way, the query we wrote shows existing relations between users on a collection of blogs. We consider that two users are related (i.e., there is a graphical link between them) as soon as one of them has a reply to another's post. In the future, we could imagine distinctions created between different networking levels (i.e., using the number of replies). As people can use different login or nicknames amongst the blogs that they use and

reply to, we used a hash of the e-mail address (using the foaf:mbox_sha1sum property) to identify users among blogs.

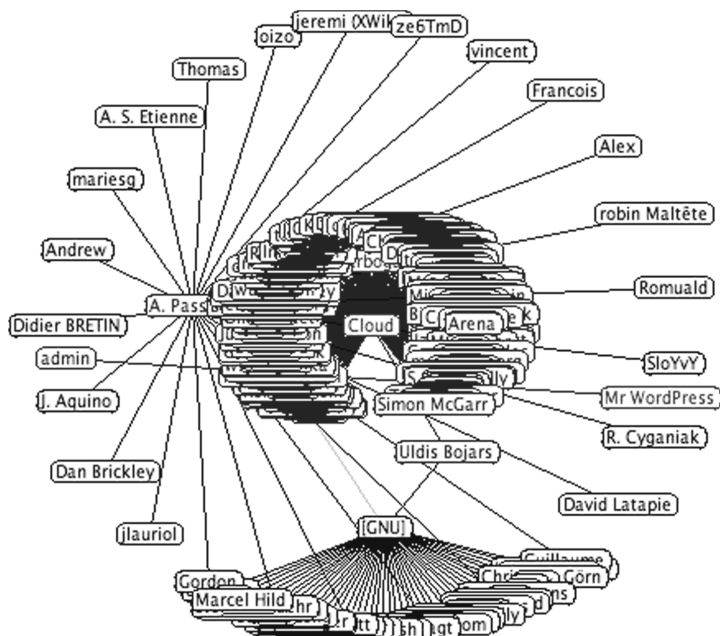


Figure 6: A network of SIOC posters and commenters

This query can displays some social networks that have emerged through a set of decentralised blog posts, and it can be useful to show people that some users may know in common, sometimes without being aware of it. While a lot of social networking services (such as orkut and MySpace) create or share communities, this SIOC query offers a nice way to see emergent communities using decentralised blogs and Semantic Web technologies. From a technical point of view, we used to Prefuse API³³ to render this network.

3.5 Use SIOC Data in Existing Interfaces

Finally, SIOC data can also be used in existing applications:

- Generic applications that can use data in their own well-defined format;
- Native Semantic Web applications that can use RDF directly.

Since we are mainly using the RDF/XML format to express SIOC data, stylesheets or translators can easily be written to render it to any other format that such generic applications can „understand“. We could also, when using the crawler mode, extract SIOC data from the store and format it according to the needs of a particular tool (we have done this for our browser implementation to make SIOC data match the Flash chart tool or Prefuse format).

As an example, we used SIMILE's TimeLine³⁴ engine (an AJAX tool for visualising time-based events) to get a historical view of our data, as shown in Figure 7. The integration of SIOC with TimeLine is described in more detail by Uldis Bojars.³⁵

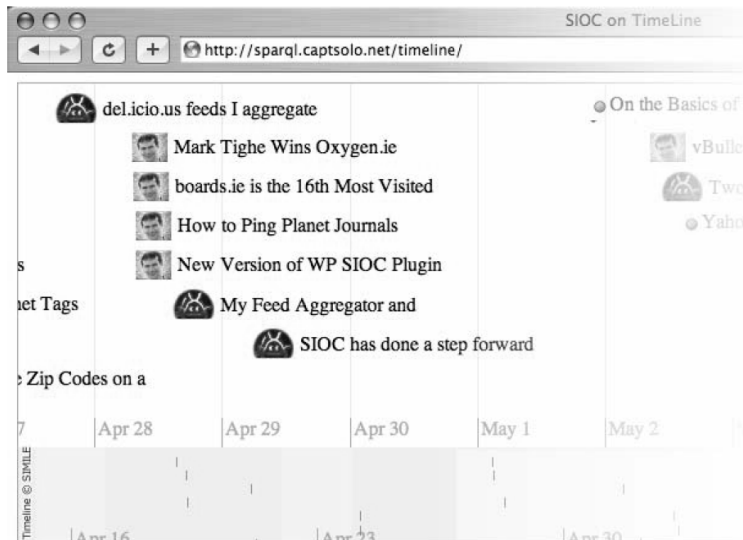


Figure 7: Using TimeLine to visualise SIOC data

The fact that a weblog's entire content, starting from its creation, is available in SIOC makes this an interesting and useful way to view such data. The challenge for using TimeLine with SIOC is that it requires live updates of the RDF store, resulting in large amounts of data and possibly leading to delays in TimeLine displaying information.

SIOC data can also be browsed using native Semantic Web applications such as MIT's Tabulator³⁶, a JavaScript service for browsing, rendering and querying online RDF data. For these applications, no translation is required as SIOC plugins are a source of pure and valid RDF - the basic content description language for the Semantic Web.

4 Future Work

We will now describe some potential future SIOC applications following on from the work presented in this paper.

4.1 Interchange Format

As we have mainly dealt with SIOC as an export format for weblogs, one aspect of our future work will consist of using SIOC as an interchange data format between platforms, in both exporting and importing data. By writing import plugins for different tools, we can easily transfer the whole set of blog data from a weblog engine to another (e.g. from WordPress to DotClear), but we could also extract `sIOC:Posts` and put them into other discussion platforms. We could thus imagine a way to push mailing list posts into a weblog engine, or into a forum, and then extract any comments and put them into another store, using SIOC data as an exchange format. This will lead to semantically well-defined connections between tools: by using SIOC as a data format to transfer information between „SIOC-ed“ platforms; and by the associated development of SIOC input APIs.

SIOC may also be used as a standard interchange format between a weblog or a collection of weblogs and tools or „widgets“ that can use such data in a meaningful

way (e.g., SIOC browsers presented earlier). This will allow to separate these tools from implementation details of any particular blogging platform and encourages to create new and interesting tools that can work with any blogging platform.

4.2 Content Negotiation

Some work has to be performed on the mirroring of post content using HTTP content negotiation. This would use existing SIOC exporters, retrieving SIOC data from the same URLs as the blog pages themselves (e.g. a browser would obtain HTML data whereas a SIOC crawler would obtain RDF data from the same URL), but there may then be some issues with the URIs that are used to identify the different concept instances.

4.3 Caches of SIOC Data

SIOC can also be used to leverage the data about blog posts that is present within aggregator „planets“ that may be used for localised communities (e.g. for a particular country) or for special interest groups. Such metadata includes blogroll links, referencing blog posts from other blog posts either implicitly or through trackbacks, tags used across communities of interest, and the most referenced external or internal hyperlinks within the community. We are also looking for other kinds of websites that would similarly benefit from exporting the information in SIOC.

By expressing this metadata using SIOC we can then use it to browse blog posts within a community in new ways, or to connect blog posts to posts or comments in other communities (through common hyperlinks, authors, tags or related tags). The network of a blog community can be visualised through `sioc:links_to` references between blogs, which illustrate mutual blogs of interest in a manner similar to „knows“ relationships in social networks. The shortest path between two blogs may also be found. There are some potential issues with metadata provision from planet aggregators. For example, when providing SIOC or re-syndicating metadata

aggregated from other sources, care must be taken to say who the cacher is and what time the post was cached at.

4.4 Future Interconnections

Finally, there are five new linkages (virtual forums, distributed conversations, unified communities, shared topics and one person, many user accounts) shown in Figure 1 that will be considered in our future work. Since the data to create these connections may be absent in blogging or forum platforms, these may be created through the use of supplemental Microformats or RDFa data in the HTML or BBCode of user signatures / profiles, post content or blog / forum descriptions.

5 Conclusions

In this paper, we introduced the different facets of SIOC, an ontology language used to describe online communities. We began by introducing SIOC in the context of the Web and online communities, and detailed ways in which SIOC can aid with the formalisation of existing links and the creation of new connections between such communities. Then, we introduced a number of different ways for creating SIOC data from existing weblogs, using exporters for different blogging platforms, and providing a PHP API so that new exporters can be easily written. We presented SIOC browsers and different ways in which one can browse SIOC information, from using an online browser to utilising a more complex architecture with crawled data, both offering new and original ways to visualise weblog data. Finally, we presented some ideas for future work on SIOC and weblogs, based on the observations and challenges detailed in our work.

References

- Adamic, L.A., O. Buyukkokten & E. Adar. 2003. A Social Network Caught in the Web. In: *First Monday*, Vol. 8, No. 6, 2003.
http://firstmonday.org/issues/issue8_6/adamic/
- Berners-Lee, T., J. Hendler & O. Lassila. 2001. *The Semantic Web*. In: Scientific American, May 2001.
- Breslin, J.G., A. Harth, U. Bojars & S. Decker. 2005. Towards Semantically-Interlinked Online Communities. In: *Proceedings of the 2nd European Semantic Web Conference (ESWC '05)*, LNCS vol. 3532, 500-514.
- Clark, K.G., L. Feigenbaum & E. Torres. 2006. *Serializing SPARQL Query Results in JSON*. W3C Working Draft.
<http://www.w3.org/2001/sw/DataAccess/json-sparql/>
- Passant, A. J.-D. Sta & P. Laublet. 2006. Folksonomies, Ontologies and Corporate Blogging. Presentation at *BlogTalk Reloaded*, 2./3. October 2006.Vienna.

-
- ¹ <http://sioc-project.org>
 - ² <http://rdfs.org/sioc/spec>
 - ³ <http://www.w3.org/TR/rdf-primer>
 - ⁴ <http://www.foaf-project.org>
 - ⁵ <http://www.atomowl.org>
 - ⁶ <http://www.w3.org/2004/02/skos>
 - ⁷ <http://esw.w3.org/topic/SIOC/PHPEXportAPI>
 - ⁸ <http://sioc-project.org/maillinglist>
 - ⁹ <http://sioc-project.org/firefox>
 - ¹⁰ <http://sioc-project.org/wordpress>
 - ¹¹ <http://semiblog.semanticweb.org>
 - ¹² <http://mu.wordpress.org>
 - ¹³ <http://sioc-project.org/dotclear>

- 14 <http://sioc-project.org/drupal>
- 15 <http://sioc-project.org/b2evolution>
- 16 <http://www.wiwiss.fu-berlin.de/suhl/bizer/rdfapi>
- 17 <http://librdf.org>
- 18 <http://sioc-project.org>
- 19 <http://www.talkdigger.com>
- 20 <http://virtuoso.openlinksw.com/wiki/main/Main/VOSIntro>
- 21 <http://sioc-project.org/firefox>
- 22 <http://pingthesemanticweb.com>
- 23 <http://esw.w3.org/topic/SIOC/EnabledSites>
- 24 <http://sioc-project.org/browser>
- 25 <http://www.w3.org/TR/rdf-sparql-query>
- 26 http://en.wikipedia.org/wiki/Ajax_%28programming%29
- 27 <http://sioc-project.org/crawler>
- 28 <http://www.w3.org/TR/rdf-sparql-protocol>
- 29 <http://joseki.sf.net>
- 30 <http://threestore.sf.net>
- 31 <http://www.w3.org/2004/02/skos>
- 32 <http://www.maani.us/charts/index.php>
- 33 <http://prefuse.org>
- 34 <http://simile.mit.edu/timeline>
- 35 http://captsolo.net/info/blog_a.php/2006/07/14/sioc_sparql_and_timeline
- 36 <http://dig.csail.mit.edu/2005/ajar/release/tabulator/0.7/tab>