Building a Connected Web Observatory
Architecture and Challenges

Thanassis Tiropanis[1], Xing Wang[1], Ramine Tinati[1], and Wendy Hall[1]

University of Southampton

Abstract. An increasing number of datasets and analytics resources have been made available on the Web. Often, those resources are hosted on portals that are related to dedicated topics (e.g. online social network analytics) or dedicated communities (e.g. Web Science researchers). A Web Observatory portal brings together communities to contribute or engage with datasets and analytic (or visualisation) applications. In addition, it provides links to dataset and analytic resources that are potentially hosted in remote locations. This paper presents the architecture of a Web Observatory at the University of Southampton, which aims to foster engagement with analytics and applications and, at the same time, support linking to remote resources in other locations, potentially hosted in other Web Observatories. We report our experiences in building the "Connected Web Observatory" and discuss open challenges both in research and standardisation.

1 Building a Web Observatory

Web Observatories aim to engage user communities with dataset and analytic resources via dedicated portals. They also aim to link to resources hosted in remote locations in other Web Observatories towards the vision of a global Web of Observatories[1]. Observatories in this sense include the following types of resources:

- *Portals;* engagement portals that bring together communities engaging with analytic projects. They enable those communities to create or share datasets and analytic applications and engage in discourse.
- *Datasets;* those provide quantitative or qualitative data, real-time data, multimedia content, open data, archives, e-Science resources.
- *Applications;* analytic applications or visualisations that relate to a specific topic or to a community.
- *Tools;* analytic resources that can support the development of datasets or of analytic applications such as visualisations. They can include harvesters when it comes to the development of datasets or data mining software when it comes to the development of analytics applications.

We envisage a Web Observatory as a socio-technical artefact. On the technological side it comprises at least a portal, which provides a list of datasets,

analytic applications or projects. Apart from a portal, a Web Observatory can also comprise data stores that host datasets and application servers that host analytic applications. Listed or hosted analytic applications can make use of one or more datasets. The tools used in a Web Observatory can include data extraction, data harvesting, dataset enrichment, visualisation tools, statistical tools or data mining tools. Figure 1 illustrates this concept of a Web Observatory as implemented at the University of Southampton. On the social side, a Web Observatory brings together people who engage with its resources and in discourse with each other, often in the context of specific projects.

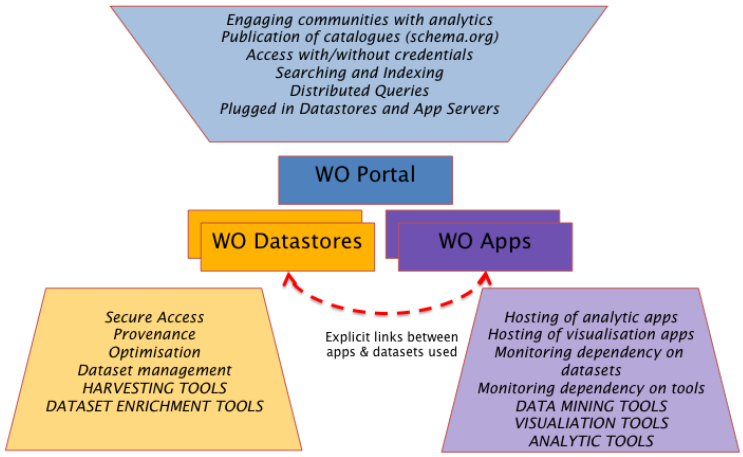

Fig. 1. Architecture of a Connected Web Observatory

1.1 Architectural Principles

Four fundamental principles were considered in the design of the Web Observatory at the University of Southampton (SUWO):

- *Not all datasets or applications can be public.* Access to some datasets needs to be restricted for licensing, privacy or other reasons. The Web Observatory allows its users to list or host datasets that are public or private. Access to private datasets is managed by the user who hosts them on the Web Observatory. Since access to datasets can be restricted, access to applications that make use of those datasets needs to be restricted as well.
- *Web Observatories list two main types of resources: datasets and analytic applications, including visualisations.* The link between a listed analytic application and the datasets that it uses must always be made explicit, even if the used datasets are listed as private, with restricted access.

– *Not all listed resources need to be locally hosted.* Listed datasets or analytic applications can be hosted in remote servers managed by third parties.
– *Metadata describing the listed resources and projects are published.* This way, descriptions of resources can be harvested and listed in other Web Observatories or Web-based resources.

1.2 The Connected Web Observatory

One of the principles in building the Southampton University Web Observatory were on making the links between analytic applications and datasets used explicit. Another one was that not all resources (datasets and applications) need to be locally hosted. Those two principles, together with the principle of publishing metadata can provide for a *Connected Web Observatory* that can serve as a building block towards the vision of a Web of Observatories[1], a global distributed resource for analytics. Figure 2 illustrates the envisaged links between portals and remote datasets, portals and analytic applications, and between such applications and datasets. *Schema.org* and the Web Observatory extension as reported in [2] is the chosen approach for publishing those metadata.

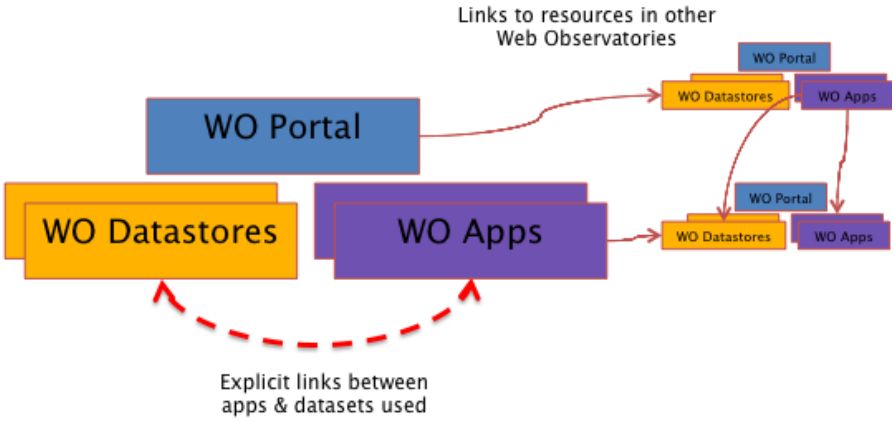

Fig. 2. Linking Connected Web Observatories

1.3 Identifying Challenges

The Web Observatory at the University of Southampton (SUWO) can link to different types of data stores such as Hadoop, MongoDB, SQL or triple stores as reported in [3] and elaborated for the case of linked data stores in [4]. It can also support different types of application servers to host analytic applications and visualisations. The requirement for access control for datasets and for

applications presents certain challenges that are discussed in section 2. In addition, enabling analytic applications to gain direct access to one or more datasets presents requirements for the support of distributed queries and for access to RESTful APIs that are also discussed in section 2. The deployment of tools for harvesting and hosting large volumes of datasets that often include real-time data present another set of challenges discussed in section 3.

2 Architecture for Sharing

To encourage community engagement the SUWO allows sharing public resources (i.e. datasets and analytic applications) as well as private ones. To enforce secure access to private resources shared on the SUWO, we deploy an access control system that hides crucial information (e.g. the URL of a remote resource) of shared resources while allowing users who have correct permission to perform certain actions.

On the SUWO two actions are allowed against a shared resource, *listing* and *accessing*. Listing gives the metadata of resources that are visible to the user. Accessing exposes the actual contents of resources. In the case of datasets that refers to querying or downloading the datasets. For visualisations that refers to displaying the visualisations. These two actions can be initiated by either users through a web interface, or applications through RESTful API. The user access control adopts a role-based strategy. The application access control (i.e. API access control) is achieved by adopting OAuth [5]. Applications authorised by a user via the OAuth server can act on the user's behalf and inherit the user's role.

2.1 User Access Control

On the SUWO two roles are defined, *user* and *owner*. Anyone who is browsing the SUWO is a user. A user can be either named (i.e. registered) or anonymous. A named user can publish some resources on the SUWO, which in turn makes the user the owner of those resources. Owners acquire new attributes such as controlling the access permission and editing metadata of resources they share. It is worth mentioning that the role is defined with respect to the relationship between a user and a shared resource. Owners only have full control of the resources they shared. They are at the same time users with respect to resources shared by other owners.

Each shared resource (dataset or visualisation) has one of the following three basic permissions, determined by its publisher:

- Public, where the shared resource is visible to and accessible by any user and application.
- Displayed, where the shared resource is visible to any user but only accessible by those who have been granted permission by the owner.
- Private, where resource the shared item is only visible to and accessible by its owner, unless the owner explicitly grant certain permission to other users.

A user can ask for access permission to an resource that is already visible to the user. The request is raised to the owner of the resource and either granted or denied. Of course, users cannot ask for listing permission for resources that are not visible.

The credentials for accessing a shared resource, such as URL, user name and password, are only maintained by the SUWO and always hidden from users. In each action initiated by a user the involved resource is referred to by a unique ID. No information of the resource can be inferred from this ID.

When a user initiates an action on a shared resource, the SUWO verifies whether the user has the corresponding permission. If positively identified, the SUWO will then execute the action and deliver back the results of the action to the user. For example, when a user browses the catalogue of resources shared on the SUWO, the catalogue is rendered specifically according to the user's permission. That is, all public resources and those are visible to the user are displayed. As a result, each user's view of the catalogue is unique.

2.2 API Access Control

A crucial feature of the SUWO is to support API access, especially API for querying datasets on the SUWO. The query API allows analytics to be easily built on top of existing data, and provides a way to better reuse existing data. The query API involves two fundamental components, an authentication and authorisation mechanism, and a set of RESTful APIs.

Authorisation using OAuth The authentication and authorisation is achieved by adopting OAuth which has been widely used to authorise applications that act on behalf of users.

OAuth defines four roles [5] as illustrated in Figure 3:

- *Resource owner*, that controls the access to a protected resource (the user of Figure 3). It is worth noticing that either a user or a owner of SUWO can be a resource owner in the OAuth scenario.
- *Resource server*, that hosts the protected resources (the SUWO server in Figure 3).
- *Client*, which is an application performing actions on behalf the resource owner and inherits its authorisation (the client in Figure 3).
- *Authorisation server*, that issues access tokens to the client if genuine credentials are provided by the resource owner (the OAuth server in Figure 3).

The OAuth flow involves three stages. Firstly the client requests authorisation from the resource owner. If positively granted, the client authenticates itself at the authorisation server and exchanges the authorisation grant with an access token. Finally, the client accesses protected resources by presenting the access token.

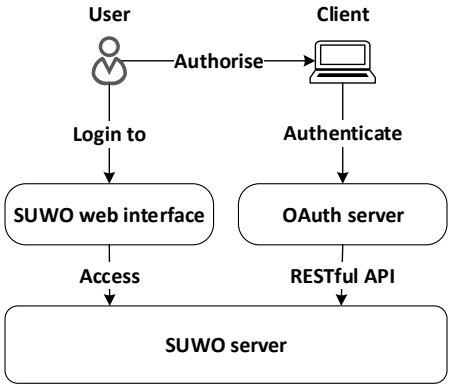

Fig. 3. OAuth Access Control of the SUWO

The RESTful API With the OAuth server in place, both public and private resources can be accessed in a secure way. At this moment the SUWO API allows clients to retrieve metadata of resources that are visible to the clients, and query datasets shared on the SUWO. The metadata of a resource include general information such as the title and the description as well as additional information for accessing the resource. If the resource is a dataset, the metadata will provide the ID of the dataset, the entry URL (which is different from the original URL of this dataset) for querying the dataset and the protocol of the dataset. For example, the protocol of a triple store is SPARQL while it is SQL for a MySQL dataset.

Querying datasets through the API follows the same flow as users querying datasets through the Web interface. The dataset is referred to via its ID and the query is redirected to the dataset by the SUWO. Once successfully executed, the result is transferred back to the client.

3 Architecture for Harvesting, and Storing

An essential component in the SUWO architecture are the data stores and related processes. Such infrastructure provides the means to harvest and store large volumes of data, including real-time streaming data, in an efficient, timely, and extensible manner. To achieve this, the SUWO draws upon the latest distributed file storage technologies in order to optimise and reduce the resources required. The design is driven by the requirement to be able to capture real-time streams of Web data, transform them into a suitable format for storage, and finally, provide a mechanism to which the data can be accessed and queried. Figure 4 provides a high level overview of the various layers that are used in the data

ingestion, storage, and access workflow. The following sections provide details on the various stages within this workflow.

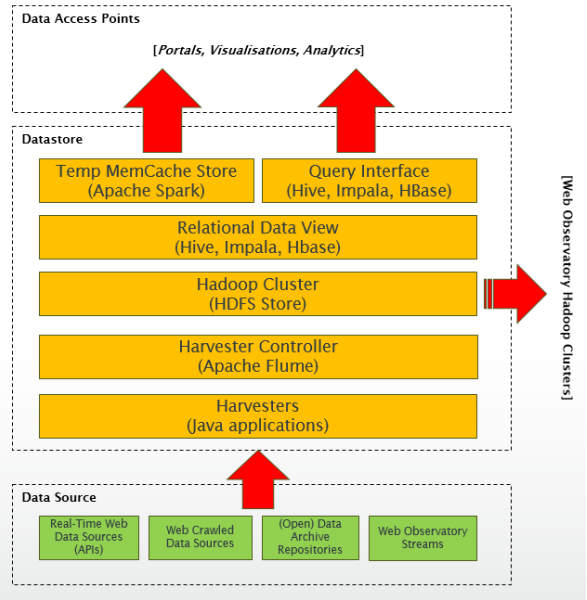

Fig. 4. Southampton Web Observatory Data store Infrastructure. Harvesting, Storage, and Hosting

3.1 Harvesting

The harvesters are developed depending on the nature of the data source intended to be captured. For the SUWO, we are harvesting a variety of Web feeds, which include real-time streams from social networking and user-generated content sites (Twitter, Facebook, YouTube), Web-crawled datasets using existing resources such as the 'CommonCrawl'[1], seeded Web crawlers, as well as other Web data repositories (which may include Open Data). In addition to these, we also plan to harvest datasets from other sources, which may include personal data stores [6], subject to individuals enabling access to their own personal datasets through discoverability mechanisms.

In the SUWO we adopt two approaches to collect Web data, via service-provided APIs, or by using Web scraping techniques. Popular services such as Twitter, Facebook and YouTube provide RESTful and streaming APIs to capture (near) real-time feeds of the data. However, due to the volume and velocity

[1] http://commoncrawl.org/

of data available, we use two approaches to harvesting data from these services; filtering by topic (or a specific entity), or collecting the whole of the data source as illustrated in [3]. Both approaches introduce challenges at various stages of the entire data ingestion to access workflow, and also have methodological implications. For Web-crawled data we also adopt similar approaches to data collection. For resources such as Wikipedia, we begin with a seeded list of pages that we wish to crawl, which the are periodically (hourly, daily, or on an ad-hoc basis) crawled. We also use general Web crawlers to gather partial accounts of the Web graph, similar to before, these are run as a batch process in order to gather temporal versions of the data.

Another resource for the SUWO is the use of data archive repositories such as those provided by the Internet Archive[2], the national archive[3], and other data repository resources such as government data repositories. Harvesting these resources is performed on an ad-hoc basis, typically with predefined criteria in order to extract a subsection of the data hosted by their respective owners.

3.2 Storage

Once data has been harvested, the SUWO uses distributed storage technologies to provide a persistent store to organise and access the data. As Figure 4 shows, we adopt the Hadoop stack in order to ingest and store the various incoming streams of harvested data. By using a distributed file system such as that offered by HDFS, we are able to process and store multiple streams of data without I/O or resource bottlenecks. At the same time, we provide for the transfer of harvested datasets in alternative formats or data stores as required (e.g. as linked data). As we are handling large streams of data, using a distributed approach allows us to scale up in terms of storage space and processing power when required [7]. We also consider that when more Web Observatory instantiations form, we will be able to share resources at the storage and processing layer, reducing the overheads of operating and maintaining large cluster infrastructure.

In order to process the harvested data streams, our approach makes use of separate processing pipelines for each of the incoming feeds. Each 'flume' handles the harvested data, and as part of this process the data is transformed into a standardised format for inserting into the HDFS file system (JSON is currently used). Whilst the datasets may include a variety of content and fields (individual communication messages, Web hyperlinks, page counts, etc.) we ensure that for each tuple, a time stamp is associated with it. Not only does this provide a means to check the current and historic operation of the harvester, but it also provides a mechanism to which datasets can be compared with each other.

3.3 Hosting and Querying

As Figure 4 illustrates, the top two layers within the SUWO data store provide access and query capability to the harvested datasets. As a consequence of us-

[2] https://archive.org/index.php
[3] http://www.nationalarchives.gov.uk/

ing a distributed file system such as HDFS, data representation mechanisms are required to view and interact with the data store. Whilst previous approaches would require extensive knowledge of MapReduce in order to access and query the data, we use various data structuring and representation approaches to enable fast and simple access to the data [7], without extensive knowledge of the MapReduce paradigm.

In addition to providing access to the data via storage query mechanisms such as Hive and HBase, we wish to provide cached access to specific volumes of data in order to provide (near) real-time access for analytical and visualisations applications. In order to achieve this we implement memory caching techniques [9], which provide rapid access to the most recent data harvested.

4 The Road Ahead

The concept of Web Observatories that can properly foster communities of people engaging with analytic resources can be powerful and promising. The deployment of a Web Observatory can be guided by architectural principles to ensure that it can support all types of resources (including private ones datasets), that it can use of resources in remote locations and that its own resources can be discovered and used by third parties. These principles can provide for *Connected* Web Observatories supporting the deployment of a global distributed resource for analytics. There are efforts on connecting to Web Observatories in other institutions engaging in Web Science and other interdisciplinary areas such as Network Science and Internet Science. Following on from the 1st International Workshop on Internet Science and Web Science Synergies at the ACM Web Science conference 2013, there will be effort on harmonisation with the Internet Science Evidence base that is developed as part of the EU-funded EINS Network of Excellence[4].

The deployment of a Connected Web Observatory presents certain challenges. Those include technological challenges on enabling secure access to datasets on different types of data stores that can be local or remote, and in enabling high performance of analytics when analytic applications rely on more than one datasets that can be hosted in different data stores. In addition, there are challenges in harvesting data from various sources in real time, in hosting large datasets and in the enrichment of those datasets for optimised performance; the latter we are going to further research in the future. The need for standardisation of metadata for Web Observatory resources is becoming increasingly critical as more Web Observatories are starting to emerge.

References

[1] Tiropanis, T., Hall, W., Shadbolt, N., de Roure, D., Contractor, N., & Hendler, J. (2013). The Web Science Observatory, 28(2), 100-104. doi:10.1109/MIS.2013.50

[4] http://www.internet-science.eu

[2] Difranzo, D., Erickson, J. S., Gloria, M. J. K. T., Luciano, J. S., McGuinness, D. L., & Hendler, J. (2014). The web observatory extension: facilitating web science collaboration through semantic markup. International World Wide Web Conferences Steering Committee. 2nd International Workshop on the Theory and Practice of Social Machines, WWW2014, 7 April 2014.

[3] Hall, W., Tiropanis, T., Tinati, R., Booth, P., Gaskell, P., & Hare, J. (2013). The Southampton University Web Observatory. 1st International workshop on Building Web Observatories, ACM Web Science 2013, 1-3 May 2013.

[4] Hall, W., Tiropanis, T., Tinati, R., Wang, X., Luczak-Rösch, M., & Simperl, E. (2014). The Web Science Observatory - The Challenges of Analytics over Distributed Linked Data Infrastructures, 96, 29-30.

[5] Hardt, D. The OAuth 2.0 Authorization Framework - RFC 6749, url:http://tools.ietf.org/html/rfc6749, October 2012.

[6] Van Kleek, M., Smith, DA., Tinati, R., O'Hara K., Hall, W., Shadbolt N. (2014). 7 Billion Home Telescopes: Observing Social Machines through Personal Data Stores,2nd International Workshop on the Theory and Practice of Social Machines, WWW2014, 7 April 2014.

[7] Pavlo, A., Paulson, E., Rasin, A., Abadi, D.J., DeWitt, D.J., Madden, S., Stonebraker, M. (2009). A comparison of approaches to large-scale data analysis. International Conference on Management of data (SIGMOD '09). 2009.

[8] Owens R.J., Femiano B. Lentz, J. (2013). Hadoop Real World Solutions Cookbook. Packt Publishing.

[9] Xin, R., Rosen, J., Zaharia, M., Franklin, M.J., Shenker, S., Stoica, I. International Conference on Management of data (SIGMOD '13). June 2013.