Archivage du contenu éphémère du Web à l'aide des flux Web[*]

Marilena Oita[†] Pierre Senellart[‡]

Résumé

Cette proposition de démonstration concerne une application d'archivage du contenu du Web à l'aide des flux Web. A partir de la spécification d'un domaine par l'utilisateur, des services spécialisés sont utilisés pour acquérir des flux pertinents. Pour chacun de ces flux, on exploite les indices sémantiques attachés à un objet dynamique pour extraire, à partir de la page Web associée, les données qui correspondent à la description. On ajoute à cet objet des méta-données supplémentaires et l'estampille temporelle, on extrait le template de la page, et on garde ces composants indépendamment pour être prêts à répondre à des requêtes temporelles et sémantiques et, à la demande, reconstruire la page Web référencée par le flux. Les méthodes pour détecter le changement de la page Web sont également utiles dans le cadre d'un crawl incrémental des versions du même objet dynamique.

Mots-clefs : archivage du Web, flux Web, objet dynamique.

1 Introduction

Innovation is constantly created on the Web through the collective intelligence of its contributors. Besides the abundance of information, we nowadays face the difficulty to keep track of updated information, as the Web is changing faster than our faculty to perceive it. For this reason, new tools are made available on the Internet in order to remain current with real-time or frequently changing data. To avoid consuming time on information gathering, a generally impacting idea was to bring content from different sources in one place, using streaming techniques. Content is made available by their publishers much easier than before, and users have only to subscribe to it, stating in this way their interest. The syndicated content has one special characteristic: it is intensively dynamic, meaning that new information is added or removed at speed rate. From news articles, blog posts, wiki entries, dictionary terms, forums entries, to any kind of event information that can be versioned, timely content is in expansion on the Web. The lifespan of the corresponding Web pages becomes much smaller than the average [1], ranging from minutes to days. The reason why we call this kind of content ephemeral is the following: when a crawler is not aware of the

[*]This work has been partially funded by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) / ERC grant Webdam, agreement 226513. http://webdam.inria.fr/

[†]INRIA Saclay – Île-de-France & Télécom ParisTech; CNRS LTCI. 46 rue Barrault, 75634 Paris Cedex 13, marilena.oita@telecom-paristech.fr

[‡]Institut Télécom; Télécom ParisTech; CNRS LTCI. 46 rue Barrault, 75634 Paris Cedex 13, pierre.senellart@telecom-paristech.fr

fresh content published, and its frequency of crawl is smaller than the content's rate of change, the respective data is lost and therefore not indexable, resulting in missing snapshots and incoherent archives.

In the Web 2.0 era, the Internet user can easily be creator of Web content, but services provided by particular sites are used in order to publish it. Moving beyond the present and into the future, the user might need to archive the published content for independent use in offline settings. Provided with a system that archives this kind of data continuously for a period of time, the user should be able to temporally and semantically query the content, and retrieve useful responses. Due to all implied challenges, ephemeral data is crawled sporadically, in an incomplete and discontinuous manner. Timely content is going to be lost if not correctly archived, and even if some information might be found and regrouped, it will be lossy, inaccessible, or fee-based. In the current *status quo* the information is stored in collections of Web pages and explored in navigational way. Hence, there is no natural way for the user to easily observe different aspects of the data and uniformly query it.

In this demonstration, we try to address these problems. The presentation is as follows: the next section gives some insights into the related work, while Section 3 explains how we acquire and utilize Web feeds to extract timely information; Section 4 reveals the importance of change detection when dealing with ephemeral content and how we can leverage the temporal and semantic aspects of the feeds to make the crawl process more coherent. A demonstration scenario and the general architecture of our system are presented in Section 5.

2 Related Work

Web archiving, seen as a necessity or duty, is acquiring a lot of importance lately due to the volatile nature of the Web, and more specifically due to the value that the lost information could have for future generations. ArchivePress, a blog-archiving project [2], is using Web feeds in order to archive pages that contain data about particular institutional activity. The principal drawback of this approach is the fact that the constructed Wordpress plug-in captures only the content that can be delivered by the Web feed. In effect, a feed can have a full coverage of the articles and their media files, but this case is quite rare: a feed is mainly just a way to advertise content. In contrast, we harness the feed's clues with the sole goal to make a better distinction between Web page components, and to extract the resulting data objects. Additionally, we do not limit ourselves to a blogging platform in particular, but aim to create a more general type of personal crawler. The preferred method for crawling is to perform snapshots periodically. While this principle is effective for relatively static pages, performing it in the case of pages for which the content changes frequently and in a heterogeneous way is leading to many problems of interpretability and temporal coherence [3]. Instead of recrawling an entire website at distant intervals of time, an incremental crawl [1] is meant to adapt its frequency of crawl to the dynamics of Web pages. This adaptation can be done only if we can determine when individual Web pages of a target site are added, updated or deleted. In absence of a prior knowledge about this change event, heuristics are commonly used [4]. A more formal model of change prediction is studied in [1]; here the authors try to determine whether changes in a Web page follow a Poisson process. However, accurately identifying the change rate of Web pages is stated by both approaches as an open problem. It is very difficult to fairly estimate the change rate when one consideres the Web page in its totality. Typically, the boilerplate that surrounds the article (dynamic ads or widgets in particular) changes the state of the DOM tree of the Web page, and, even if the content of interest remains the same, the Web page is seen by a naïve archiving crawler as changed. Trying to filter this boilerplate can be risky, because we might not manage to cover all kind of structuring techniques used. Instead, we extract only the element of interest (generally called data object) and detect content change at this finer-grained level. Extracting structured data from Web pages has been intensively studied. However, none of the existing works takes into account the semantic of the content in order to extract it, but instead harness patterns or code regularities at DOM level. Our approach is somehow similar to [5] in our way of detecting the parent that has the largest number of children that satisfy a certain condition. In contrast, the condition to be satisfied for us is not the similarity of encoding at data-record level, but instead a semantic measure at the level of conceptual nodes. This semantic measure is evaluated by matching concepts (from an initial bag of semantic clues given by the feed) against DOM node elements of the Web page in which the identified data object resides. We have compared the article's text content returned by our technique with the result of *Boilerpipe* [6]. Boilerpipe extracts articles from Web pages using the text density ratios of subsequent blocks. There are situations in which this technique fails either because the text of the article is too small and a denser portion of the page is identified, or because there are various articles in the page, and no semantic distinction between them can be done: all of them are taken as one. By exploiting the semantics of a Web feed, we can obtain more precise results. However, our method works only for Web pages that have a feed associated.

3 Capturing Data Objects

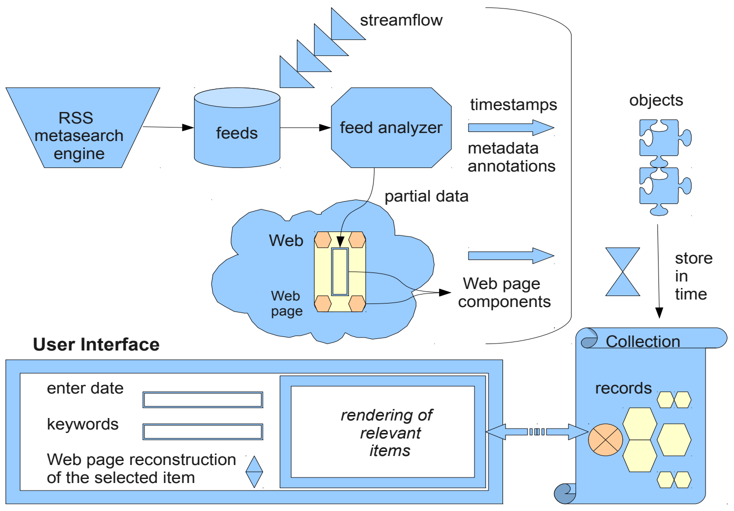
To help readers keep current with dynamic content, publishers add feed capabilities to their dynamic resources. We use the concept of data object to refer to a news article, blog post, dictionary entry, a forum message, a status, or any other kind of resource that is uniquely referenced by a Web feed item. In order to address the problem of the abundance of feed sources, we pass through a discovery phase that is also performing a semantic filtering. We use a specialized service like *Search4RSS* metasearch engine to acquire a number of Web feeds in a specific domain, the scope of which is specified by a user with keywords. Search4RSS covers all kind of Web pages that have a feed associated, in RSS, Atom or Rdf formats. We begin by identifying the data object that corresponds at feed level with an item of a channel. The item's URL references the Web page that will be cleaned for further processing using HtmlCleaner. We use the hints provided by the feed item's title and description to construct a bag of descriptive terms and *n*-grams. This semantic information about the data object that we are looking to extract will be used to identify the DOM node element that acts like a wrapper for the text and references of the respective Web article. Finally, the data object will be set with the extracted text and references, and timestamped with the item's publication date. The concepts retrieved from the feed item by means of data mining will be associated to the object in order for it to be semantically queried. Similarly, the object's timestamp will be exploited for the temporal queries. We separately crawl other semantic components of the page and the remaining template. References to these are retained as extrinsic attributes of the data object, with the goal of reconstructing at need the Web page, or to make some kind of relational analysis. As well, we enrich the current semantic context of the data object with a related ontological one. The detection of tag clouds and categories of the Web article can be done using special heuristics, but when this metadata is missing we make use of the YAGO ontology [7] to retrieve instances and concepts related to the article's semantics.

4 Change Detection

While in this demonstration the main aim is to add exploitable semantics to Web archiving collections, the feeds and in particular the data object extraction technique can also be used to improve the Web page change detection, an essential step in the context of an incremental crawl. Assuming that the website to be crawl has a feed associated, a temporal analysis of the feed can be done periodically. The resulting website's publishing strategy can be leveraged to automatically get the feed files with an adequated frequency and use them as an instrument for change detection. Useful timestamps can be extracted from a Web feed, like the `pubDate` RSS element or `updated` Atom element. These temporal hints give the dates of publication of the referenced Web articles and from our experiments are, with very few exceptions, omnipresent in a feed. An incremental crawl can therefore use them as change detection signals in order to crawl new pages of a particular site. Basically, the feed's channel presents a dynamic part of the website it refers to. Hence, in theory and by excluding boilerplate, other pages of the site are relatively static and a periodic snapshot can capture their content without losing information. For pages discovered through the feed and which correspond to feed items, we can empirically set the frequency of change by detecting the modifications at data object level.

5 Scenario

To better illustrate the motivation, we can take the example of an ecologist who wants to study the evolution of a natural process and its parameters of influence. She knows that there exists *on the Web* services that can provide data of interest in a timely manner. Her interest is consistent in time: she wants not only to get informed for a period of time about the states of the same process, to get different points of view of the same event, but also to be able to make some analysis at the end of a specific cycle. The ecologist wants also to browse the information of interest offline, and to query it using temporal attributes and keywords. The actor in this scenario could simply use the feeds provided by webmasters to get informed of new Web pages of interest, but actually she will have no ability to manage and query particular data from those Web pages in a uniform manner. Our system provides the environment and atomic data necessary to this process, as it is shown in the following figure.



Given the user's domain of interest, we query a Search4RSS metasearch engine and accumulate a number of feeds as references to Web pages that provide timely content. Partial data in the above figure represent the resource's title and description. By having quite a clue about the content that is advertised through the feed's items (considered therefore content of interest), we are able to identify in the Web page the data objects. The user has the possibility to follow a channel and the potentially many versions of the same data object, or to search for other sources on the same topic (reiterating the request to *Search4RSS*). As well, the user can run semantic and temporal queries and the system will provide her the relevant data objects in the form of references to their authentic, initial form: the snapshot Web page (or a cleaned version of it). As a future improvement, the system can use users' preferences to render the information, more specifically to mix the objects or present them in different contexts.

6 Conclusions and Observations

Ephemeral content is the object of many timely sources on the Web today. Its value is stated by the existence of Web feeds. An incremental crawl is adapted to this kind of content, but when the Web page changes too rapidly for the crawler to detect the change, some pieces of the puzzle that represent content in time will be missing. The demonstrated system presents a new way of archiving dynamic data, an approach based on feed technology. In essence, we exploit the hints provided by the Web feeds to encapsulate timely content together with metadata into a self-contained timely unit. This will represent an instance in the chain of updates of the same Web resource and also the *diff* element necessary in the context of an incremental crawl. The goal of our application is to add exploitable meaning to Web archiving collections created in a more complete and coherent manner.

References

[1] Junghoo Cho and Hector Garcia-Molina. The evolution of the Web and implications for an incremental crawler. In *Proc. VLDB*, Cairo, Egypt, September 2000.

[2] Maureen Pennock and Richard Davis. ArchivePress: A really simple solution to archiving blog content. In *Proc. iPRES*, San Francisco, USA, 2009.

[3] Marc Spaniol, Dimitar Denev, Arturas Mazeika, and Gerhard Weikum. Catch me if you can. Temporal coherence of Web archives. In *Proc. IWAW*, Aarhus, Denmark, September 2008.

[4] Kristinn Sigurðsson. Incremental crawling with Heritrix. In *Proc. IWAW*, Vienna, Austria, September 2005.

[5] Bing Liu, Robert Grossman, and Yanhong Zhai. Mining data records in Web pages. In *Proc. KDD*, Washington, USA, August 2003.

[6] Christian Kholschutter, Peter Fankhauser, and Wolfgang Nejdi. Boilerplate detection using shallow text features. In *Proc. WSDM*, New York, USA, February 2010.

[7] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A core of semantic knowledge unifying WordNet and Wikipedia. In *Proc. WWW*, Banff, Canada, May 2007.