Delivering new web content reusing remote and heterogeneous sites. A DOM-based approach.

Luis Álvarez Sabucedo
Universidade de Vigo
Galicia, Spain
lsabucedo@det.uvigo.es

Luis Anido Rifón
Universidade de Vigo
Galicia, Spain
lanido@det.uvigo.es

ABSTRACT

This contribution addresses the development of new web sites reusing already existing contents from external sources. Unlike common links to other resources, which retrieves the whole resource, we propose an approach where partial retrieval is possible: the unit for data reuse is a node in a DOM tree. This solution permits the partial reuse of external and heterogeneous web contents with no need for client (browser) modifications and just minor changes for web servers.

Categories and Subject Descriptors: H.4.3 [Information Systems]:Miscellaneous[Communications applications] H.5.4 [Information Systems]:Information interfaces and presentation[Hypertext/Hypermedia] I.7 [Document and text processing]:Electronic Publishing

General Terms: HTTP Interoperability

Keywords: HTTP Web Server Interoperability Reusability URL DOM Content reuse Hypertext.

1. INTRODUCTION

During the last few years, a research trend in Web technologies is related to information retrieval, interoperability and content reuse. We propose the development of a server side solution that provides agent users with capacity to access contents generated in a collage fashion. Using this solution we will be able to reuse contents in a standard way regardless of particular technology used to create them.

The main goal for this project is to provide a cost-effective solution to recover partially documents and therefore, be able to compose new web contents. Our proposal complies with the following requirements: a server-side simple support for partial data retrieval in web environments. In order to achieve this goal we propose the use of the URL[2] schema with minor extensions.

Thus, we can compose *lively* new pages from contents recollected all over the World Wide Web, i.e., there is not intermediate steps, contents are gathered in a fully automatic fashion to compose new contents. It is important to bear in mind that pages are made of several already existing nodes, no matter where they come from. It is possible to include contents created dynamically from any DBMS by using any technology, static contents, local contents or what ever that can be accessed as a DOM tree.

2. CONTRIBUTION

As all web pages may be defined in terms of its DOM tree[1], we can *re-make* new web contents just by mixing nodes from different web resources. In order to do this, we just need to be able to address nodes in the DOM tree remotely from the web server responsible for content delivering to the client. With this idea in mind, when a user agent requests a web page from a server, this server must collect nodes from those servers where contents are actually stored and compose the final HTML resource that must be provided in response to the user agent. This method of generating contents works properly as any web content can be rendered from and to a DOM tree.

To achieve this goal we need to perform two extensions in current web servers:

- Extend the requested URL format to express nodes in a DOM tree.
- Insert a module on web server to collect just the involved data.

So far, when a client requests a web resource, no matter if this is an HTML page, a flash element or an image; it just sends an URL to ask for the desired resource by its name as a file in its own filesystem. In our proposal we need to be able to express an element from the DOM tree in the requested resource. We may consider this new requested condition as a refinement of the current format.

At first sight, we could think of an already in-use solution such as Xpath[8]. We dismiss this option due to the high level on complexity that would introduce in further phases of the project.

We decide to make use of similar notation to anchors in HTML, i.e., $file.html\#node1$. Thus, we chose the following format:

```
http://server.org/resource.xhtml//node1
```

By using this format, we mean that the desired resource is $node1$, an element in the file $resource.xhtml$ located in the server $server.org$. The main reasons to choose this format are:

- This schema for URLs fits with the current specifications provided in the RFC about URL[2].
- It is quite simple to integrate it in already developed software.

WWW 2005, May 10–14, 2005, Chiba, Japan.
ACM 1-59593-051-5/05/0005.

- As far as we can determine the resource with no possible misunderstanding, we are able to know what is the real request.
- Most of already working web servers are able to manage this pattern to address contents.

Instead of finding the right file and submitting it, our web server must deal with an additional issue: parsing the file and gathering the requested information. Our server must get the file and process it until it just gets the desired piece of information, the addressed node. The implementation of this behaviour does not require a large amount of resources and for the prototype that has been developed the chosen library was the Jaxen library[7]. The rest of the process is the same for any other webserver: send the information through a HTTP channel.

It is important to note that clients do not notice how the content delivered is composed since the information received is just a HTML with no intervention from our platform. All data transformations from the original HTML-like file are completed in the server so no updates or upgrades have to be done in the client side.

3. ALTERNATIVES

As previously stated, interoperability issues are not a new matter to deal with. Over the last decade a lot of resources were devoted to overcome the current situation and, also, currently many ongoing projects are being developing to evolve interoperability and reusability of contents. We will present a brief review of the most insightful projects and alternatives that fulfil a similar function.

RSS feed. By using RSS[4], systems are able to share contents by interchanging a XML[3] document. This document, compliant with an already existing standard, provides information about news and events. This file must be created and maintained and wide range of information is not supported.

Dynamic contents. Web servers can generate contents on-the-fly depending on user agent requests. There are plenty of options to create this kind of contents: CGIs, PHP, ASP, .NET platform, JSP, ... These technologies could be useful to fulfil our goal but there are several limitations: we must install an engine to execute those source codes, those must be programmed for each case and we should agree on a particular protocol to send and receive data if we are looking for certain pieces of documents. Of course, it would not support legacy information systems as you should update contents themselves, or at least introduce a programming addition, to meet requirements.

Management Reporting System(MRS).We can also find other solutions in the market such as Web Squirrel[6] or Hunter Gatherer[5]. Drawbacks for the use of this options are the too large level of complexity they introduce and also the functions provided are do not exactly match our needs. So these solutions may not be suitable for our environment.

4. CONCLUSIONS AND FUTURE WORK

The main idea of this contribution is to propose an alternative way to reuse contents in a cost-effective way. As the only needed changes are on to the software installed in web serves, neither in the information itself nor changes in client side, this technique can spread quickly with nearly no efforts in the short term and supporting legacy information systems.

To increase the usability of this proposal, it would be useful to support the use of the *src* attribute in more HTML tags. By using the current version of XHTML just few tags, such as *frame* or *img*, can support this technique. Also, we will work on providing a common look-and-feel for generated web contents. Nevertheless, and taking into account the testing phase, we can state that, after a short period of training, non-expert web masters are in position to create contents using our proposal.

The overall result is that we will be able to provide several advantages for creating new contents from heterogeneous sites, even from dynamic contents, with no changes on the client side and without the need for the introduction of additional logic on web server.

5. ACKNOWLEDGMENT

We want to thank “Ministerio de Educación y Ciencia” for their partial support to this work under grant “Metodologías, Arquitecturas y Lenguajes para la creación de servicios adaptativos para E-Learning - MetaLearn” (TIN2004-08367-C02-01).

6. REFERENCES

[1] Mozilla. Gecko dom reference. Web available, Feb 2005. http://www.mozilla.org/docs/dom/domref/.

[2] Network Working Group. Rfc 1738 - uniform resource locators (url). Web available, 2005. http://www.faqs.org/rfcs/rfc1738.html.

[3] RSS-DEV Working Group. extensible markup language. Web available, 2005. http://www.w3.org/XML/.

[4] RSS-DEV Working Group. Rdf summary site. Web available, 2005. http://web.resource.org/rss/1.0/spec.

[5] M. schraefel, Y. Zhu, D. Modjeska, D. Wigdor, and S. Zhao. Hunter gatherer: Interaction support for the creation and management of within-web-page collections. In *11th World Wide Web Conference*, 2002.

[6] R. M. Simpson. Experiences with web squirrel: My life on the information farm, 2001.

[7] The Werken Company. jaxen: universal java xpath engine. Web available, 2005. http://jaxen.codehaus.org/.

[8] W3C. Xml path language. Web available, Feb 2005. http://www.w3.org/TR/xpath.