Overview of Extratos: Yet Another Service Oriented Information Extraction System for the Web*

Andrés Sanoja, Claudia León

Universidad Central de Venezuela, Facultad de Ciencias, Escuela de Computación, Centro de Computación Paralela y Distribuida, Caracas, Venezuela

andres.sanoja@ciens.ucv.ve, claudia.leon@ciens.ucv.ve

October 4, 2008

Abstract

This article describes the design and implementation of Extratos, a Service Oriented Information Extraction System for web content sharing, based on web services as extractors and BPEL business process generation. Some insights from archaeological sciences are applied to the design of the system. It is organized in five subsystems: Xpathula, Lab, Node, Web Portal and Executor and the external entities web browser, web page and orchestration engine. Our solution present a web extraction process, from the perspective of users and software systems, with three phases: design, generation and execution. The goal of the design phase is to help the user “discover” text from web pages, transform them in Text References, conform Pages and compose a Mashup, and the corresponding extraction procedure. The goal of the generation phase is, based on design, is convert the mashups as the result of a service oriented process. For the execution of the process is given a service oriented infrastructure to allow access to software clients through to mashups, using web services standard protocols.

Keywords: mashup, web extraction, mashup generation, BPEL, web services, web services orchestration, distributed systems.

1 Introduction

Now a days sharing content has become in a very common task for users, subscription to sources using RSS (Really Simple Syndication), Atom Feeds, web services in general or simple XML documents are a few examples. Other techniques are focused on letting users to select text directly from web pages and store it into repositories (i.e. Google Notebook)²; social utilities encourage users to share, using tagging or using characteristics of a social network.

---

1This work is supported by the FONACIT project “Acercamiento de la Tecnología al Ciudadano” No. 2005000166 and UCV

1http://ccpd.ciens.ucv.ve/extratos

2http://www.google.com/notebook/
These content available from the sources can be processed using tools, which assists users to accomplish the task of compose information; this enable users to generate web content as the result of combining others, from different sources. These kind of applications are referred as Mashup. Examples of this type of tools are Dapper.net $^3$ and Yahoo Pipes $^4$. The mashup concept is discussed in [23], they clearly state that, generally, mashup tools has to deal with the following issues: source modeling, data clearing, data integration and data Visualization. Keeping the same meaning but using a different terminology we may add that these basic issues can be named: design, generation and execution of an extraction process, which are more like to the one exposed in [20]; we think of these concepts as phases of the whole process; later on we’ll explore these concepts more in detail.

Web extraction is a well-investigated subject; great deal of research has been done. TSIMMIS$^8$ is a collection of tools for building extractors$^5$, was focused mainly to assists developers to realize and systematize the extraction. Other specialized tools that assists developers are the query languages for the web. This languages allows to realize queries similar to databases, but for web pages, for example: Florid [21] and Lorel [3]. In the other hand there are implementations oriented to users interaction, for example OLERA [7], Dapper.net and Marmite [25], generalizing, they are based on mashup creation and management.

Other aspect are the automatic extractors, these type of solution are focused on machine learning and inference; a few examples are: RoadRunner [11] and KnowItAll[15]. There are, also, service oriented implementations using web services as extractors, as the presented in [18]. Service Oriented Architectures has been used in conjunctions with web services to implement extraction processes as [5, 6, 17] and [20]. Having an architecture like these, which implements an extraction process, allows developers to get different configuration of a system, and also a certain degree of freedom to adjust it to certain requirements, but may be more complex due to the inherent distribution.

In the recent years the amount of web content available has been continually growing, mostly as result of human activity. Even when using web design tools, web page generators or others kind of techniques for web content creation, there is participation of human users. More users are including the information found in web pages into text documents, spreadsheets, presentations and software tools, as well. Users have to deal with some characteristic of the web, for example: location of the web page, language, aesthetic, passwords, forms, etc. Software developers use the same kind of information, but other characteristics are relevant, for example: internal representation of web page document (HTML, XML, plain text, XHTML), malformed documents, character codification, pattern matching, etc. In this context, both users and developers have in common the web extraction$^1$. From the user perspective it is realized, usually, as a “copy and paste” task, where text from one document is placed into another. Software developers use other kind of techniques such as scrapers, extractors, web wrappers, spiders, agents, Information Extraction Systems, among others [22, 12].

The extraction activity is not just a concept in computing and informatics, is a well-investigated subject on disciplines such as chemistry, engineering and archaeology, for example. Each approach, are intended to solve it using diverse methods and techniques. During the initial phase of the research we tried to explore, informally, the coincidences between the computing approach and contrast it with others, in this case: the archaeological prospection. The main goal is to look for insights that might be useful for designing Information Extraction Systems.

In this article is presented an overview of Extratos, yet another Service Oriented Information Extraction System for the web. Is intended to allow users to share content in the form of mashups.

$^3$http://www.dapper.net
$^4$http://pipes.yahoo.com
$^5$We use the term extractor in a general way to describe software tools which allows to obtain information from web documents
composed by text selected directly from web pages. Gives a services oriented infrastructure to extraction activities, mashup generation, visualization of results to users and to software clients, as well. Explore some insights from the archeology field, describing the extraction in this context and using some of them in the design of the system. This implementation is the result of an ongoing research. In section 2 we presents some aspects of extraction in archaeology and the web, then in 3 the solution and in 4 we conclude.

One of the main goals of the implementation is that the system must be configurable and distributed, and using web services standards and protocols. We have the hypothesis that using this technological approach, will permit us get different instances of the same system, which gives us the opportunity to explore different topologies, results and requirements.

2 Archaeology, Extraction and the Web

In this scientific discipline there exists proven theories about extraction, methods, techniques, tools (instruments), case of studies, and is the result from formal research and empirical experiences from the beginning of XX century. Following the scientific spirit, we ask ourselves: is there a correspondence about extraction in archeology and computer science?, can we use some of this experience in the design of an Information Extraction Systems?. On the first look, although it seems to be incompatible, theory and methodologies used in both disciplines strive to reach a very related goal: to get information from material and digital artifacts that are the results of human activities.

According to Vargas [4], Childe [10] and Dunnel [14], in archaeological research the geographical location of the artifacts in space, which is relevant and essential to define the spatial and chronological context of the archaeological fact, is identified by coding standards. This kind of location is named: site. An example of site is an urban city location (like Caracas, capital of Venezuela) where the code is DC4, which corresponds to the 4th site in Capital District (Distrito Capital); this code is unique and should not be used to identify any other site. This site is divided into sections and identified by a number. So DC4-1, is the 1st section of the DC4 site. As the excavation is taking place in the different sections and artifacts and other archaeological traits are exposed, it is relevant and necessary to order the time context, to record the level (depth) and the spatial context where they were extracted. A level may be arbitrary and its scale may be variable and measured in metric units. Levels may correspond also with the layers of a cultural or a natural deposition. Continuing with the example, and artifact can be tagged to be found in DC4-1-5, which means the 5th level of the 1st section on the DC4 site.

When the excavation phase has been completed, begins the classification of all the items collected in the site. Artifacts are grouped according to a certain previous criteria: level of collection, type of material, form, function and associational contexts. Using different observational techniques and methods, artifacts are identified and classified by established criteria and ordered in sets: for instance if they belong to a certain age of history, according to the material they are made of, their form and function, their spatial and/or temporal association, etc. The utilization of these criteria depends on the theoretical-methodological background of the scientist and the aims of his research.

Excavation and classification has no meaning themselves, but part of a scientific process where the goals, methodologies, methods, techniques, tools and policies are defined; the techniques of extraction must be guided by sound theories and methods, since they are crucial for scientific validation of information extracted from its context.

In the context of the World Wide Web are used the terms web page that identifies a web document, which resides in a more complex structure called web site, which is a collection of related web pages.
Moreover documents are organized in levels, known as Superficial and the Deep Web[19].

The web page concept is similar to the concept of artifact, and the web site with the concept site used in archaeological prospection. Is interested to see how these contextualization is valid to identify the artifacts and how to extract them from their natural context, moreover, the classification activities give to these artifacts a more complex one. Each object is located into a taxonomy, in the case of archeology it depends on the expertise of the researcher. In the computer sciences there are several techniques that allows us to accomplish the same task: bookmarks, tags, folksonomies, etc., which one to use?, also depends on the researcher or developer.

For an archaeologist, after an object has been extracted, is well known the interest he/she has on its properties (materials, form, etc), which becomes attributes to the object in study. In the web could be similar, after an text has been extracted we are interested on the text format, character coding, etc. but at this level of research we can not assure that with a piece of text, inference can be made to determine or classify the document and know its properties, using software tools.

One of the most relevant information obtained after the classification in archaeology is a taxonomy and the correspondence with each object. In the web is similar when a mashup is created, sometimes mashup are related to a template, or are under a simple relationship page-block, and surely many others.

More coincidences can be observed, as in archaeology to each web site is given an IP address and, in general, a domain name, which should not be, used to identify any other web site. Table 1 presents a comparison of some terms in both disciplines [24].

<table>
<thead>
<tr>
<th>Archaeology</th>
<th>The Web</th>
<th>Archaeology</th>
<th>The Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site</td>
<td>Web Site</td>
<td>Collection</td>
<td>Mashup</td>
</tr>
<tr>
<td>Artifact, Object</td>
<td>Page</td>
<td>Surface Prospection</td>
<td>Superficial Web</td>
</tr>
<tr>
<td>Fragment</td>
<td>Text</td>
<td>Subsoil Prospection</td>
<td>Deep Web</td>
</tr>
<tr>
<td>Site Code</td>
<td>DNS, IP</td>
<td>Taxonomy</td>
<td>Bookmarks, Tags, Directories</td>
</tr>
<tr>
<td>Section</td>
<td>Path, QueryString</td>
<td>Classification y Sorting</td>
<td>Semantic Classification</td>
</tr>
<tr>
<td>Level</td>
<td>Xpath</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Extratos, a Fragment is the text extracted and its reference, and the reference it is realized as a 6-tuple (uri, xpath, r1, r2, userid, timestamp), where uri is the localization of the document, xpath is the localization of the text element within the document, r1 and r2 are limits of the text range of the selection, and userid represents the user who create the fragment. An Artifact is a set of Fragments and a Collection is set of Artifacts where exists a hierarchical relationship: collection ⇒ artifact ⇒ fragment

Some comments about the relationship between terms: an artifact could be just a fragment, a collection could contain only one artifact; the same for the web: a page could be just one text, the mashup could be just one page. Is clear that these are exceptions but are valid.

2.1 Extraction

By nature, manually intervention in archaeology has as result a permanet alteration of the landscape and the site itself. Can not be repeated more than once for the same section of a site, and of course
for the same artifact. That is why is of such importance the expertise of the researcher; only has one
shot. Moreover, even after a successful extraction of an artifact should be identified in the context, taking note of the site, section, level and some insights which assist researches when have to analyze it in the laboratory[9].

In the web the extraction activity can be done more than once, systematically without alter the context of the text, subject to extraction. However, the web has a strong dependence on the underlying infrastructure and others non technicals factors, as well. Technically, Internet could be a non reliable communication channel, therefore messages can be lost, duplicated, logical and physical issue, etc. Other non technical factor has to be considered, such as the temporal aspects of the web content, for example: web sites that are shutdown because lack of payment, hosting companies that leave bussines, web sites redesigned, content changing, etc. One may think, in the worse case, that the system was able to do just one successful extraction before the site went down. Thinking in this context we can look forward and may say that the goal is to have the most recent copy of the text and, if possible, keep it updated. This approach is very much alike to one possible extraction as in archaeology, of course is a particular case.

2.2 Process

Childe describe the field work activity in archaeology. While reading one can highlight some expressions that evidence “steps” or activities that occurs during excavation and extraction of artifacts. The author does not list them explicitly, but we interpret the main ideas. Here are some of them, which we think might be the most relevant an enumerate them in Table 2. Besides them are our interpretation or the closest equivalent in the web.

<table>
<thead>
<tr>
<th>Archaeology</th>
<th>The Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>indicate the spot where to dig</td>
<td>Text selection, Xpath determination</td>
</tr>
<tr>
<td>after discovery an artifact, clean it up to re-</td>
<td>check if the document is well formed</td>
</tr>
<tr>
<td>reveal it as much as possible</td>
<td></td>
</tr>
<tr>
<td>carefully remove it from the soil</td>
<td>using the Xpath inspect the DOM tree and</td>
</tr>
<tr>
<td></td>
<td>copy the text(s) node(s)</td>
</tr>
<tr>
<td>assign a code (following the coding standards)</td>
<td>Reference: URL, Xpath, Date</td>
</tr>
<tr>
<td>store in a bag</td>
<td>Store in a database</td>
</tr>
<tr>
<td>in the lab, analyze and classify it.</td>
<td>Mashup</td>
</tr>
</tbody>
</table>

Following this approach we think of an extraction process for the web as the conjunction of these main archaeological activities: prospection and classification (also known as Field Survey and Analysis). After seeing that indeed there are similarities in the concepts, activities and interest, think to include some of this archaeological concepts in our work, shown in the Table 1, for design of the system and use the web terms for the implementation. The design is guided using software development methodologies and methods[16], but using the semantic of the activities mentioned above.
3 Solution

In this section is described the implementation of Extratos, the terms mentioned before are used (collection, artifact and fragment) but applying their equivalent in the context of the web (see Table 1).

Extratos, generally speaking, is intended to assist users to create mashups (collections) from sources where information is semistructured, such as web pages, and provides an infrastructure to support the generation and visualization. Others source formats are not taking into account at this level of the research (RSS, databases, documents, web services, etc). As mentioned earlier, we divide the extraction process in three phases: design, generation and execution. In design phase users pick up text (fragments) directly from web pages in a web site and store them in Extratos, then are able to create and manage pages (artifacts) and mashups. A page is organized following a hierarchy of the type: \textit{mashup} $\implies$ \textit{page} $\implies$ \textit{text}. This solution handle not only the last updated value of the selected text, but also the reference which describe how to get it. On demand of a user a page and mashups can be generated, which is convert the references, mashups and pages defined into descriptors, and process them. The system is updated with: an service oriented specification of the process and extractors to obtain the text. This process can then be used by any client application, invoking the process and obtaining a XML document result of consulting all the reference indicated, and organized following the established hierarchy. An user can visualize the information using as client the system, presented in various formats. In the Figure 1 is shown the architecture of the system.

![Figure 1: Extratos Architecture](image)

It is organized in five subsystems or applications: Xpathula, Lab, Node, Executor and Web portal, the entities web browser, web page and orchestration engine, two databases: cache and references. Xpathula application is intended to assists users in text selection. The goal of the Lab application is to serve Xpathula (an others client applications) and provide an user interface for managing mashups, and has the responsibility to control generation activity. The role of the Node application is extraction and updating (refreshing) references; also has a role in the generation activity. The Executor application has the responsibility of control the execution of a process and serve as an wrapper to the orchestration engine; has a process specification in BPEL\(^6\) and participate in the generation ac-

\(^6\)Business Process Execution Language
tivity. The Web Portal is a web application which is intended to allow users to share content. The orchestration engine is a component that allows interpreting and executing BPEL processes.

There are several activities that take place at different time. In figure 2 is presented the general activities or functionalities of the system and when they took place.

The figure 2(a) shows the interaction between components in the design phase. Using the browser an user select and pick up text directly from the web page currently viewed, using the application Xpathula (a Mozilla Firefox Extension). The main goal is to determine the text that surround or is near the user click. After inspecting the DOM (Document Object Model) is obtained an Xpath expression. The user is prompted to select a range of the selected text (perhaps is interested in just a number or a date, and not the entire paragraph). The Xpath, URL (Uniform Resource Locator), the range and the user identification conforms a reference.

A mashup can be manage by the user: create new one, modify and existing, delete or rename it. When a mashup is created is empty, then the user can add new areas to associate new pages. The user can also manage pages supporting the same create, modify, delete and rename operations. In an empty page a user can associate text references, also can define text replacements and text operations. These (replacements and operations) take place after the text is extracted or updated. Replacements are simple String operations that seek for a pattern and replace it with a new one. Operations are simple string or arithmetic operation that can be used with text values; replacements
occurs before the Operations. By now, the syntax used for replacements are regular expressions, and for the operations, parameterized statements using Ruby programming language syntax. For example:

\[
\text{Substitution("54,123.00","/","\") = "54123.00"}
\]

\[
[\text{Reference}_1].to_f + [\text{Reference}_2].to_f
\]

In the figure 2(b) can be observed the applications that interacts to generate an extraction process. The generator component in the Lab application prepare descriptor for the process and for each extractor, and send it to the Executor and Nodes. In the same order, descriptors correspond to the mashup, page and text. On the arrival of messages, each Node update the Reference database with the reference data (URL, Xpath, \( r_1 \), \( r_2 \), userid and timestamp). Attempt to extract the text from the web page, if successful update the Cache database with the text value, after apply replacements and operations, if any. For each reference, so far, is scheduled an update every 24 hours. Each node holds the global variables \textit{amountofextractors} and \textit{activity}. The first one count the total extractors generate in a Node, and the second is a counter which increase its value when a extractor is invoked. Before sending descriptors to Nodes is collected these values from all Nodes and the messages are sent to the Node which has less activity and less extractors.

When the Executor application receives the message take the process descriptor and produce a set of files necessary to create a BPR (BPEL Process Archive)\(^7\), in order to deploy it in the orchestration engine. The files created are: WSDL of web service extractor and main process, a PDD file (Process Deployment Descriptor), a Catalog and the BPEL source document. All this files are compressed into a BPEL Process Archive (BPR) which is copied into the deployment folder of the Engine.

The figure 2(c) show the interaction of components necessary to conform the XML document. On a request the Executor application invokes to the Orchestration engine, internally in the BPEL process document are specified the necessary calls to the Nodes. This activity is done using standards and protocols of web services (SOAP and WSDL). On each request, the Nodes query the Cache databases and look for the text value, previously processed, and give it on return. In the case of reference could not be update or extracted previously an error messages is included in the response.

When all the Nodes has been consulted and the Engine finished processing the BPEL process, the resulting XML document is sent as response to the client. As part of this first version of the system the Lab application handle some presentation formats for the user: HTML tables, SQL, PDF, CSV (Comma Separated Values), Excel or plain XML; to software clients is sent a SOAP response directly from the Executor application.

In the next section is presented an example in order to test the functionalities of the system.

### 3.1 Example

The example presented is to construct a mashup with the following information:

- get the 2007 population of France from the web page

- get the 1999 population of France from the web page
  \[http://en.wikipedia.org/wiki/Demographics_of_France\]

\(^7\)http://www.activevos.com/
get a descriptive text of the Metropolitan France area from http://en.wikipedia.org/wiki/France

calculate the absolute difference in the population values

Using the Xpathula application the text where selected and sent to the Lab. In figure 3 is shown a screenshot of the Xpathula application when collecting the 2007 France population. The same procedure was done for the 1999 population and the descriptive area text.

Using the Lab application two pages were created: France Population Example and France Area. The first one is composed by the 2007 and 1999 population values, while the second is composed only by the descriptive text. To the first page replacements and operations were defined. Because the original text values are [FP1999]=“58,519,000” for 1999 and [FP2007]=“61 875 822” for 2007, was needed to replace in the first one “,” by “” in the second one “ ” by “”. Then a subtraction operation was defined and is [FP2007].to_f - [FP1999].to_f and should be equal to 3356822.00.

A new mashup “France Info” was created. The two, previously created, pages were associated. In Figure 4 is show the user interface for mashup composition, showing the to pages associated.

When the mashup were generated the extractors were updated and the BPEL specification deployed. Through the Lab application send an execution request and get the XML document. The HTML table presentation format was applied. In figure 5(a) is shown and extract of the XML document and in figure 5(b) is a preview of the HTML presentation.

The process performance for this example were measured: the whole refereces update took 1.73 seconds and the process execution 4 miliseconds. This execution was done in three computers: two running a Node application each, and the other with the Lab and all others components of the system. The Node’s computer configuration was Intel Dual Core processor at 1.60Ghz and 1GB of RAM memory, the other configuration was Intel Pentium 4 running at 1.80Ghz and 512Mb of RAM memory. The programing languages used are Ruby 1.8.7 and OpenJava 1.6.0. The operating systems used are GNU/Linux running the 2.6.18 kernel version and the Nodes with Ubuntu Linux running on the 2.6.27 kernel version.
4 Conclusions

As mentioned in the introduction, this article describes an Overview of Extratos. At this research phase we have covered all the initial requirements and have a functional tool. However, there is still a lot of work to do. Has been an interesting experience explore extraction from a different perspective, from other disciplines point of view. Initially it was just an experiment and an exercise of observation, but is clear that is a valid approach. Although the main insight used in this work was the form that artifacts are extracted and a way to organize them, many others activities and methods can be explored. What is more interesting to us, as starting point, are the classification and analysis methods. There are different form of organizing the data and how to interpret it, that we think worthy of review for instance: seriation, how to determine the social context of an artifact, when was created, by whom (cultural group, not individuals), properties; artifacts sites evolution and conservation of heritage; has a potential application in the web, and in fact much of them has already its corresponding.

Because the design was influenced by the archaeology method, this tool has still a lot work to do to improve the way of extracting text. Archaeologist extract one artifact at a time (we mean
one person) but, in fact in the web multiple text can be extracted at the same time using pattern matching, web languages to realize queries or querying by example.

The response times of extractors where observed but not formally measured, though this informal observation allows us to see the strengths and weakness of the web extractors and how they can be improved in a future work.

Some experiments has been conducted to improve the performance of extractors for updating text. Using a fixed 24 hours refresh rate is a good way to start, but we think the correct approach is to assign to every page different levels of importance and differentiate the content from presentation. A newspaper web site is expected to change faster than a personal web page, for instance. Many web pages has content mixed with “noise” like advertising, banners and scripts, that may give a false positive on change detection. Using statistical distributions to estimate the time of change based on the web page type, seems to be a good starting point. In fact these experiments are based on prior experience by the LIP6 (Laboratoire d’Informatique de Paris 6) whom has assumed it as a research topic[13] and between our two universities have been shared experiences.

Has been positive to have a running service oriented implementation. The low coupling between components allowed us to observe the potential and flexibility for its use into an organization. The implementation has minimal differences from the design specification, this is one of the benefits that gives us the methodology and technology chosen. To configure the system it was not necessary to modify the source code of applications, just customizing parameters and values were needed.

Use a workflow language for specify the process let us work in a high level of detail and give a better understanding of the extraction. Is planning to explore other technologies for workflow specification, been the one to start the ActiveXML Project [2]. One of the insight we like most of this project is the simple syntax and concept. BPEL is a good standard and the engines have good performances, however the syntax is complex in comparison with AXML’s.

To conclude is important to mention that this project not only has produced a software system, also is important the experience gained in this two years, as well as two publications and a master degree thesis.

References


