A Practical Approach to Modeling and Extracting Information from Semantic Web Based on Microformats*

J. Guadalupe Ramos, Ricardo A. Solís, Héctor Oceguera
Depto. de Sistemas y Computación
Instituto Tecnológico de La Piedad
Av. Tecnológico 2000, CP59300, La Piedad, Mich. México.
{guadalupe@dsic.upv.es, solis.itlp@gmail.com, hos6509@hotmail.com}

Josep Silva
Depto. de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n, E-46022, Valencia, Spain.
jsilva@dsic.upv.es

***Abstract*—The lowercase semantic web consists of web pages enriched with semantic special tags which are called microformats, and it is considered a pragmatic path to the Semantic Web. In this work, we present a practical approach for modeling (microformat based) semantic relations between web pages by means of classical graph like data structures, such as semantic networks. In order to provide categorization into the semantic network we implement the special set of entrance points to the semantic network, which are so-called, semantic indexes. Then we present an agent software approach to retrieve semantically related information between web pages, we describe the main modules and data structures of the process and finally we present snapshots of the tool.**

***Keywords*-microformat; lowercase semantic web; semantic network;**

I. INTRODUCTION

Often, when a user requires valuable web based information in order to provide support for his decision-making process, she is forced to read and properly analyze every web page by searching the specific and meaningful data. Considering the growing size of the web (on March 2009 the indexable web had at least 25.21 billion pages [1]) this task becomes tiresome and time consuming.

Task automatization of information is a particular activity which is frequently delegated to software agents. Indeed, we expect a software agent to act on behalf of someone to carry out a particular task which has been delegated to it [2].

Unfortunately, a web oriented software agent needs advanced features, namely semantic preparation of the web pages, which is still not profusely present in the web world. Nevertheless, there are notable efforts in the web engineering field pursuing the semantic development of web content.

In this work we introduce a software agent approach in order to retrieve meaningful units of information from web content based on particular simple semantic annotations which are present in sets of web pages on the Semantic World Wide Web. This proposal can reduce the amount of web pages that a user should consider when she is performing a web search.

The Semantic Web is considered an evolving extension of the World Wide Web in which the semantics of information and services on the web is made explicit by adding metadata. Metadata provides the web content with descriptions, meaning and inter-relations. The Semantic Web is envisioned as a universal medium for data, information, and knowledge exchange [3].

Many important technologies for developing the Semantic Web are already in use, for instance: The *eXtensible Markup Language* (XML), the *Resource Description Framework* (RDF) and the *Ontology Web Language* (OWL) among others [4]. All of them share the goal of providing common vocabulary and complete standard languages in order to be employed in the web page constructing by the broad and heterogeneous set of web developers. In this setting, each web developer should learn and use those languages to incorporate semantic annotations and specifications in their web projects.

Nevertheless, efforts to extend the Web with meaning (with the aforementioned tools) have gained little traction. These initiatives have been bogged down by complexity and over-ambitious goals, or have simply been too much trouble to implement at a large scale (see, e.g., the discussion in [5]).

Perhaps, the completeness and complexity of the standard semantic technologies contrast with the simplicity of the first fundamental languages for web page preparing like, for instance, the *HyperText Markup Language*, HTML.

Recently, a new initiative has emerged that looks for attaching semantic data to web pages by using simple extensions of the standard tags currently used for web formatting

* This work has been partially supported by the Spanish *Ministerio de Ciencia e Innovación* under grant TIN2008-06622-C03-02, by the *Generalitat Valenciana* under grant ACOMP/2009/017, by the *Universidad Politécnica de Valencia* (Programs PAID-05-08 and PAID-06-08) and by the Mexican *Dirección General de Educación Superior Tecnológica* (Programs *CICT 2008* and *CICT 2009*).

in (X)HTML[1], these extensions are called *microformats* [6], [7]. A microformat is basically an open standard formatting code that specifies a set of attribute descriptors to be used with a set of typical tags.

Example 1: Consider the following XHTML code that introduces information of a scientific event.

```
<h2>Scientific Events</h2>
<p>
  Date:
  <span title="2009-09-21">
     September 21th, 2009
  </span>
  <p>
     Mexican International Conference
     on Computer Science (MICCS'09)
  </p>
     ITESM, Mexico City <br />
     It promotes the publication of scientific
     results of the international community
     related to applied and fundamental
     research on Computer Science <br />
</p>
```

Now, let us see the same web page code but taking into account the inclusion of the standard `hCalendar` microformat [8], which is useful for representing information of scheduled events.

```
<h2>Scientific Events</h2>
<div class="vevent">
 Date:
  <span title="2009-09-21" class="dtstart">
    September 21th, 2009
  </span>
 <span class="summary">
    Mexican International Conference on
    Computer Science (MICCS'09)
 </span>
 <span class="location">
     ITESM, Mexico City
 </span>
 <div class="description">
   It promotes the publication of scientific
   results of the international community
   related to applied and fundamental
   research on Computer Science
 </div>
</div>
```

The `class` property qualifies each type of attribute which is defined by the `hCalendar` microformat. The code starts with the required main class `vevent` and classifies the information with a set of classes which are auto-explicative: `dtstart` indicates starting date, `location`, information where the event will be and so on.

Microformats are a clever adaptation of semantic XHTML that makes it easier to publish, index, and extract semi-structured information like tags, calendar entries, contact information, and reviews on the web. See [6] for a complete list of information entities capable to be annotated.

[1]XHTML is a sound selection because it enforces a well-structured format.

Microformats have given rise to the so-called *lowercase semantic web* [9]. In the rest of the paper we discuss about lowercase semantic web w.r.t. Semantic Web (in capital letters) and we refer particularly to the set of standard microformats into web pages that compose it. Indeed, microformats are considered a pragmatic path towards achieving the vision set forth for the Semantic Web [7].

Recently, in the semantic web setting [10] has proposed a formal approach, the use of *semantic networks* which is a convenient simple model for representing semantic data. A semantic network is often used as a form of knowledge representation; and it is formalized as a graph whose vertices represent concepts, and whose edges represent semantic relations between the concepts [11].

The approach of [10] is based on an extension of semantic network which they call *indexed* semantic network, this notion of semantic network contains indexes that allow us to extract information chains which are related to a specific kind of semantic information.

Unfortunately, in [10] there is not a mention of an experimental tool, neither design, to demonstrate the usefulness of their theoretical notions.

Motivated by these ideas, we introduce a software agent approach for searching and extracting microformats which are present in sets of web pages. Then, they are modeled towards indexed semantic networks; furthermore a report is shown to be consumed by a final user.

The rest of the paper is organized as follows. In Section II, we overview the topic of semantic networks and recall the basic concepts related to them. In Section III, we describe how semantic networks can be built from the semantic web. Next, in Section IV we describe some of the practical aspects of our approach: data structures, main processes and algorithms related to the tools developed. A running example is introduced in Section V, where we emphasize the practical advantage of using our agent approach. Finally, we conclude in Section VI.

II. Semantic Networks

The concept of *semantic network* is fairly old—in fact, the term of semantic network dates back to Ross Quillian's works [12] where he introduced it as a way of talking about the organization of human semantic memory—in the literature of cognitive science and artificial intelligence. Nevertheless, it is a common structure for knowledge representation, which is useful in modern and different problems of artificial intelligence. For instance, in the recent Semantic Network Analysis Workshops [13], [14] many applications of this formalism were discussed, e.g., for social networks or hypertext networks.

A semantic network is a directed graph consisting of nodes which represent *concepts* and edges which represent *semantic relations* between the concepts. Sowa [15], [11] introduced a classification of semantic networks, in which

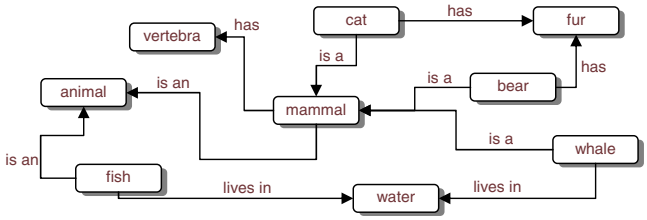

Figure 1. A definitional semantic network.

the type of *definitional networks* emphasizes the subtype of *is-a* relation between a concept type and a newly defined subtype. This is the kind of semantic network that we will use in this paper. In Figure 1, we present a typical example.

III. Modeling lowercase semantic web in semantic networks

In this section we introduce the notions for modeling the lowercase semantic web in semantic networks, in addition we present the formal representation for semantic network (see [10] for a detailed explanation).

A. Modeling semantic web pages

A web page is composed of multiple kind of labels, this approach focus on microformat classes as convenient entities for modeling, and then, for indexing or referencing. Labels for formatting text as: `<strong>text</strong>` are ignored, since they do not offer semantic information.

Example 2: Let us consider again the semantic microformatted web page code of Example 1. The semantic information is classified by using predefined classes which can embed other classes. For instance the main class `vevent` embeds the `summary` class (to introduce a brief text explaining about an event), the `description` class (to provide a long description of events), etc.

Now, let us see the next code which shows a semantic web page composed by two main classes, i.e., `vevent` and `vcard` (for people, company, organization or place card information microformatting [16]):

```
<h2>Scientific Events</h2>
<div class="vevent">
 Date:
  <span title="2009-11-09" class="dtstart">
     November 9th, 2009.
  </span>
 <span class="summary">
    8th Mexican International Conference
    on Artificial Intelligence, (MICAI 2009).
 </span>
 <span class="location">
    CIMAT, Guanajuato, Gto., México.
 </span>
 <div class="description">
   The aim is to bring together leading
   researchers from all over the world,
   interested in advancing the state of
   the art in Artificial Intelligence.
  </div>
</div>


<h2>Staff</h2>
<div class="vcard">
 <span class="fn">
   <strong>
     Arturo Hernández Aguirre PhD
   </strong>
 </span>
   <p class="role">
     Organization Chair
   </p>
   <div class="org">
     Center for Research in Mathematics (CIMAT).
   </div>
  <div class="adr">
     <div class="street-address">
        Callejón de Jalisco s/n,
        Mineral de Valenciana
     </div>
     <span class="locality">
        Guanajuato, Gto., México
     </span>,
     <span class="postal-code">
        36240
     </span>
  </div>
  <div class="tel">
     01 (473) 73 50800 ext. 49657
  </div>
 </div>
```

Example 3: Consider again the microformatted code of Examples 1 and 2. From their classes we can build the semantic network depicted in Figure 2 (the grey parts of the figure do not belong to the semantic network and thus they can be ignored for the time being).

In the figure, the nodes of the first page are labeled with $P1$ and the nodes of the second page are labeled with $P2$. Thus, nodes (i.e., classes) are unique. We observe two kinds of edges: The `locality` class from Example 2 is embedded in the `adr` class. Thus, there is an embedding relationship from node `adr` to node `locality`. Furthermore, `vevent` in $P1$ and `vevent` in $P2$ of the semantic web of Example 2 are linked by a semantic relationship since they are the same kind of class. Observe that we add to the graph two additional concepts, $P1$ and $P2$, which refer to web pages. This is very useful in practice in order to make explicit the embedding relation between microformats and their web page containers.

The set of modeled relations allows us to locate where the microformats are and which are their related classes.

B. Formal presentation of semantic networks

In this subsection we present the formal notions related to semantic networks for semantic web [10].

Definition 4 (semantic network): A directed graph is an ordered pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is a finite set of vertices or nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of ordered pairs $(v \rightarrow v')$ with $v, v' \in \mathcal{V}$ called edges. A semantic network is a directed

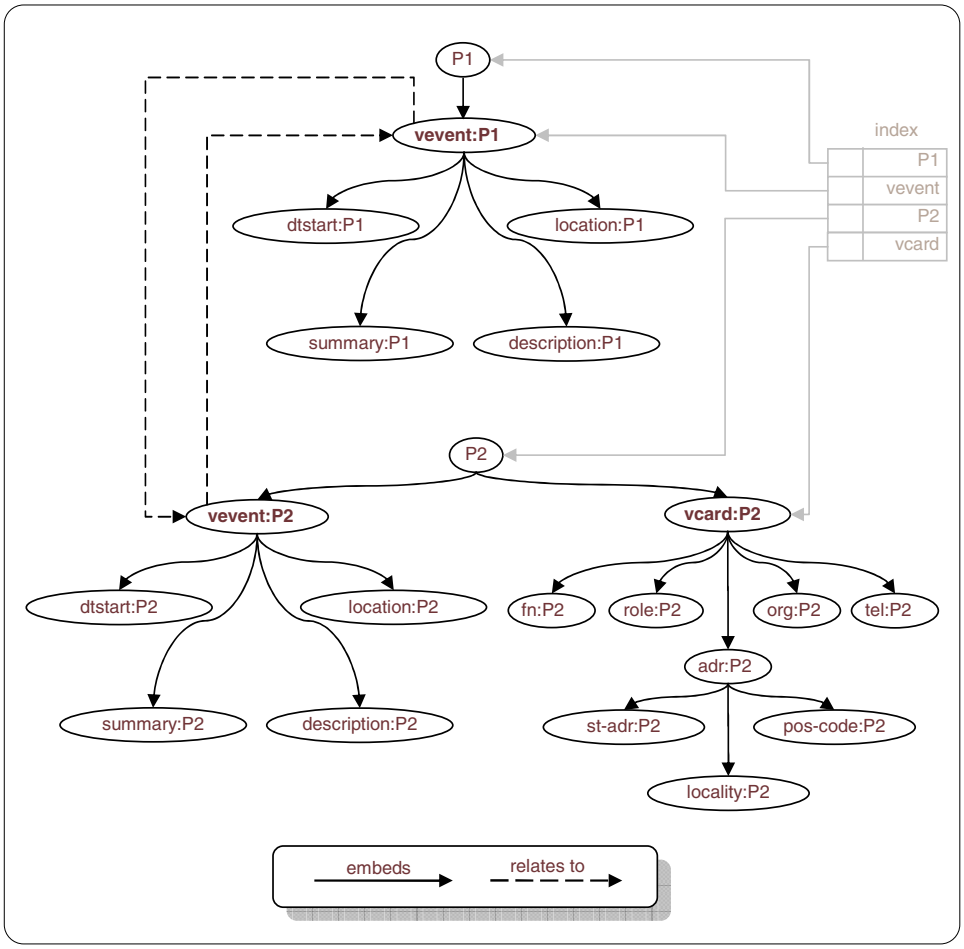

Figure 2. Semantic network of Example 1 and Example 2.

graph $\mathcal{S} = (\mathcal{V}, \mathcal{E})$ in which nodes have been labeled with names of web pages and microformatting classes of these pages.

As an example of semantic network consider the directed graph in Figure 2 (omitting the grey parts) where nodes are the set of microformatted classes provided by two semantic web pages.

A semantic network is a profuse mesh of information. For this reason, the semantic network is extended with an *index*. The index contains the subset of concepts that are relevant (or also visible) from outside the semantic net. Each element of the index contains a key concept and a pointer to its associated node. Artificial concepts such as webpages (See $P1$ and $P2$ in Figure 2) can also be indexed.

Let $\mathcal{K}$ be a set of concepts represented in the semantic network $\mathcal{S} = (\mathcal{V}, \mathcal{E})$. Then, $rnode : (\mathcal{S}, k) \rightarrow \mathcal{V}$ where $k \in \mathcal{K}$ (for the sake of clarity, in the following we will refer to k as the *key concept*) is a mapping from concepts to nodes; i.e., given a semantic network $\mathcal{S}$ and a key concept k, then $rnode(\mathcal{S}, k)$ returns the node $v \in \mathcal{V}$ associated to k.

Definition 5 (semantic index): Given a semantic network $\mathcal{S} = (\mathcal{V}, \mathcal{E})$ and an alphabet of concepts $\mathcal{K}$, a semantic index $\mathcal{I}$ for $\mathcal{S}$ and $\mathcal{K}$ is any set $\mathcal{I} = \{(k, p) \mid k \in \mathcal{K}$ and p is a mapping from k to $rnode(\mathcal{S}, k)\}$

Now, the index extension for semantic network.

Definition 6 (indexed semantic network): An indexed semantic network IS is a triple $IS = (\mathcal{V}, \mathcal{E}, \mathcal{I})$, such that $\mathcal{I}$ is a semantic index for the semantic network $\mathcal{S} = (\mathcal{V}, \mathcal{E})$.

Example 7: The semantic network of Figure 2 has been converted to an IS by defining the index with four entries $P1$ (page1.html), $P2$ (page2.html), $vcard$ and $vevent$. Thus, for instance, $vevent$ entry points to the cycle of $vevent$ nodes.

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and two nodes $v_1, v_n \in \mathcal{V}$, if there is a sequence $v_1, v_2, \ldots, v_n$ of nodes in $\mathcal{G}$ where $(v_i, v_{i+1}) \in \mathcal{E}$ for $1 \leq i \leq n-1$, then we say that there is a *path* from v_1 to v_n in $\mathcal{G}$. Given $u, v \in \mathcal{V}$ we say that the node v is *reachable* from u if there is a path from *u* to *v*.

Definition 8 (semantic relationship): Given a semantic network $\mathcal{S} = (\mathcal{V}, \mathcal{E})$ and a node $v \in \mathcal{V}$, the semantic relationships of v are the edges $\{v \rightarrow v' \in \mathcal{E}\}$. v is semantically related to a concept u if there exists a semantic relationship $(u \rightarrow v)$.

Given a node n in a semantic network, we often use the term *semantically reachable* to denote the set of nodes which are reachable from n through semantic relationships. Clearly, semantic reachability is a transitive relation.

IV. Tools for processing semantic networks from lowercase semantic web

In this section we focus on the practical approach for semantic relations discovering and for microformat extraction from sets of web pages, and thus from the semantic web.

Certainly, there are notable efforts to extract microformats from web pages [17], and to filter HTML documents [18]; however current approaches only focus on single web pages, and thus, they ignore the relations between data which is located in different web pages. One of the reasons to consider isolated analysis of web pages is the required time to download the code of many web pages. We consider an approach slightly different, i.e., a software agent w.r.t. a browser add-on of [17], [18]. An agent allows us to download a set of web pages and then we are able to analyze their (X)HTML code.

We developed a pair of tools: a *semantic relation searcher*, and a *semantic analyzer*. The goal of the first tool is to discover semantic relationships and make a report. In this way, we can launch a sample of URL's and determine which are the best web sites (those that contain microformats) in order to be later profusely analyzed. This first tool is faster than the second one. The second tool takes an URL sample an extracts the microformats found in the visited web pages.

A. Semantic relation searcher

The process that the semantic relation searcher agent performs is depicted in Figure 3.

The process is composed by the following main phases and data structures:

Web page searcher: For the semantic analysis a set of web pages is required, for this, we develop a web page for web searching. The web page uses the Google's Web Search Server which is queried by means of the Google AJAX Search API [19]. Thus, a query to Google is launched, and then we filter the Google's response in order to obtain only URL's. URL's represent the sample to study. The web page searcher (`Searcher.htm`) was allocated in the `http://www.dsic.upv.es/~guadalupe/` site and its parcial code is as follows:

```
function OnLoad() {
 var searchControl =
   new google.search.SearchControl();
 searchControl.setResultSetSize(
   google.search.Search.LARGE_RESULTSET);

 var searcher =
   new google.search.WebSearch();
 searcher.setUserDefinedLabel(
   "Web Analyzer Results ...");

 var options  = new
   google.search.SearcherOptions();
 options.setExpandMode(google.search.
   SearchControl.EXPAND_MODE_OPEN);

 searchControl.addSearcher(searcher,options);
 searchControl.draw(
   document.getElementById("searchcontrol"));
 searchControl.setSearchCompleteCallback(
   this, searchComplete);
}
```

Roughly speaking, we create a search control box: (for Google queries) `searchControl`, which is configurated to ask eight answers by page from Google. Then we create a searcher object (`searcher`) which will use only the Web Search Server, no images, no videos search, etc. Some options are defined, for instance, the results will be presented in expanded mode (title and resume lines). Furthermore, the searcher and options object are linked to the search control in the web page (`searchControl.addSearcher(searcher, options)`). The web page object where the results will appear is established by means of `searchControl.draw`, and finally the method `searchComplete` will be activated when a search is completed. `searchComplete` is used for filtering URL's which are useful towards sample preparing.

Sample: It is a data structure. Once a query is performed, we pick the URL results up, and prepare the sample to be analyzed. The sample is a list of certain number of URL's, i.e., the web pages to be browsed and then analyzed.

Loading and extracting (X)HTML: For each URL in the sample, we navigate to it and then, we extract its (X)HTML code in order to search semantic entities, i.e., microformats. For this, we employ the C++ Builder's CppWebBrowser component.

Microformat searcher: This module searches the main classes of microformats, e.g., `vevent` string. Once the string is found, then the algorithm verifies the validity of the microformat by searching that the microformat string be preceded by the `class=` substring, and that the complete label `<label ... class = "vevent" ...>` be well formed (i.e., no comment, no closing HTML label and simple text neither).

Since web page developing is not a compiled process (as typical programming language developing is), it is common to find web pages with class annotations slightly different. Here the performance of an automaton is useful, for this, we deploy a push down automaton: PDA, see the Figure 4. A PDA incorporates a stack in order to memorize read characters. A PDA transition is obtained from the current state, the input character of an analyzed text and a stack string and produces a next state and, sometimes, an updated stack.

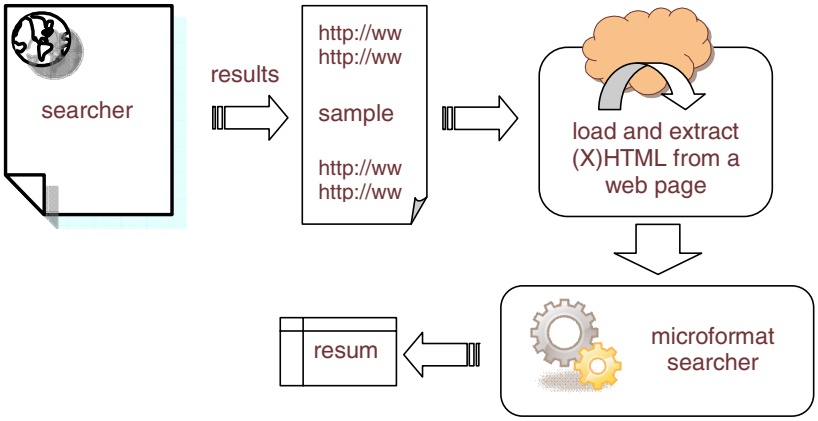

Figure 3. The process for semantic relation discovering.

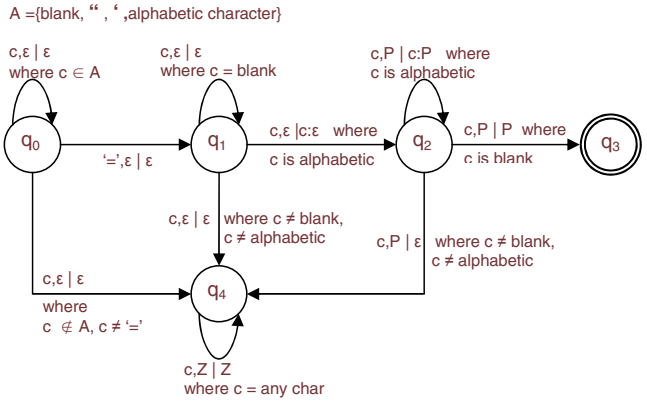

Figure 4. Push-down automaton for discovering of microformat classes.

Our PDA changes its state according to the character read and in some cases the character is pushed to the PDA stack. The PDA is composed by the $\{q_0, \ldots, q_4\}$ states. ϵ means an empty stack.

The automaton is positioned in the previous character of the microformat string. Then it follows reading from the right to the left until it retrieves the symbol =. Now we should find the substring `ssalc`, i.e., the reverse of class. For this, in the automaton, when we are in the state q_1 and if we read an alphabetic character then we move to the state q_2, the character is pushed to the stack. Finally if a blank is read we go to the acceptance state. In the program code, when we end in the acceptance state we verify that the stack content is equal to `ssalc` substring and then, as a consequence, we can state that we have a microformat.

The Microformat searcher process does not extract microformats, they are only identified.

```
if (token ==
      "< label ... class = ' vevent ' ... >")
   {
       veventMfCount= ++
       microformatName = vevent + veventMfCount
                                  +":"+ wepPageName;
       return microformatName;
   }
```

In order to provide unique microformat names we take the web page name, a consecutive number and the type of that specific microformat.

Resum: It is a data structure in matrix form. Its goal is allow to an user to choose a specific web page and a particular type of microformat, and then the name of microformats found will be presented.

B. Semantic analyzer

The process performed by the semantic analyzer agent is depicted in Figure 5.

The process is composed by the main phases and data structures described below. Firstly, the *web page searcher*, the *sample* and the *Loading and extracting (X)HTML* step are similar to the previous tool.

Semantic Analysis: The semantic analysis is a traversing process into the code of a web page. The process searches the string of a microformat, e.g., the `vevent` string. Once the string is found, the algorithm verifies the validity of the microformat annotation by checking that the microformat string is preceded by the `class=` substring (by employing the automaton of Figure 4), and that the complete label `<... class = "vevent"...>` is well formed.

```
if (token ==
    "< label ... class = 'vevent' ... >")
 {
   opener = label;
   open = 1;
   microformat = token;
 while(open > 0 )
 {
   read(token);
   if(isValidToken(token))  // no comment
   {
     microformat = microformat + token;
```

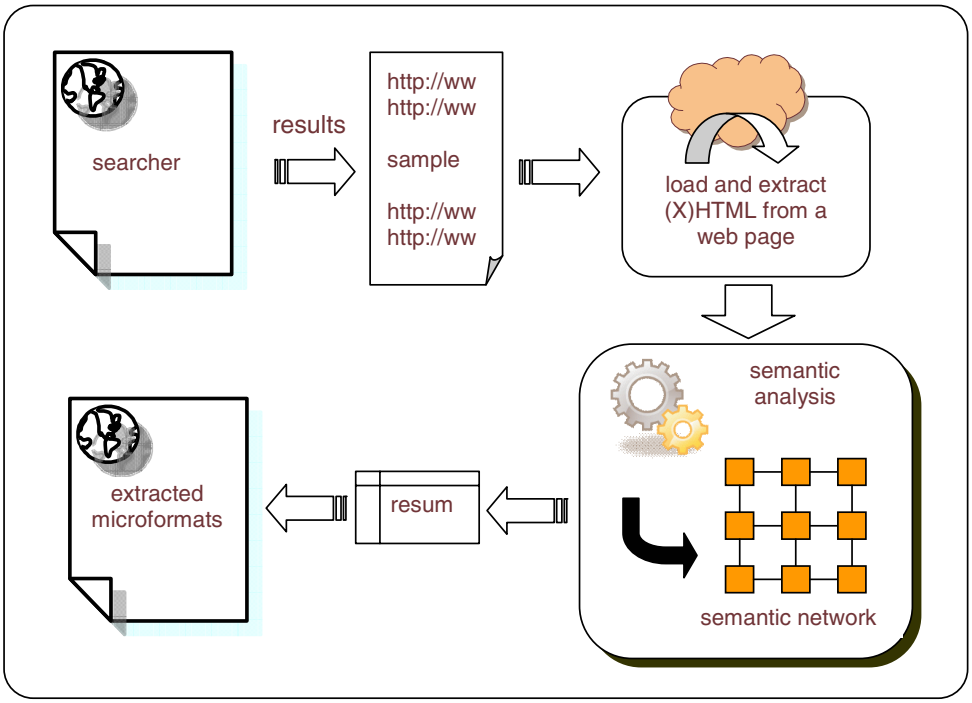

Figure 5. The process for microformats extraction.

```
    if ( openingToken(token) and
         labelToken(token) == opener)
         open++;
    else if (closingToken(token) and
             labelToken(token) == opener)
             open-;
   }
  }
 }
return token;
```

When a valid token is detected, we take the label name where the microformat is contained and then the variable `open` is incremented. Basically, `isValidToken(token)` is developed as automatons that verify positively the following cases:

- An opening label `<label...>`.
- A closing label `</label...>`.
- A self closing label `<label.../>`.
- A simple text content.

Note the no inclusion of comments which are discarded. When a closing label is found an its label name is the same that the opener label, then the variable `open` is decremented. The process to retrieve the complete microformat is performed while `open` is greater than `zero`.

Semantic network: The semantic network is a data structure which is employed to compose the semantic relations. The semantic network can be viewed as a matrix of sets of strings (i.e., microformat code) where the horizontal dimension is composed by the microformats that own to a particular URL. The vertical dimension traverses and relates those microformats present in different web pages and that share a particular class of microformat. For instance, each column can be viewed as the set of microformats of a type, e.g., `vevent`, that are located in the set of web pages. Therefore, any cell is a set of a particular microformat type that belongs to a specific web page. Now we could think in a cubic shape of the matrix (see Figure 6).

Now let us make a mapping between the semantic networks presented in Section III for semantic web and the data structure employed in the implementation.

An *indexed semantic network* requires a set of entrance points to the semantic network, i.e. the indexes, in Figure 6, they are `vevent`, `vcard` and `geo`. This changes according to the user selection for microformat searching.

The *semantic network* is composed by nodes and edges, the nodes are each one of the complete microformats, since only the microformat main classes were considered in this implementation. Observe that there are two types of semantic relations between nodes, embedding and relating (see Figure 6). Regarding the data structure, the horizontal dimension provides embedding relations and the vertical dimension provides the relationships. Thus there is, at least, a path from each page, e.g., from P1 to each microformat embedded and other from the index, to the microformat of that specific type. Therefore, each node is semantically reachable by its web page container (embedding) and by its

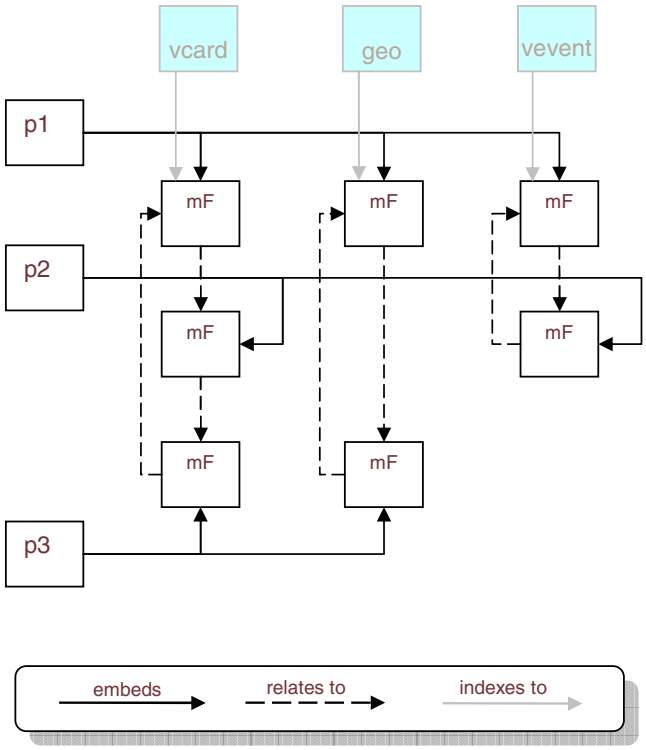

Figure 6. Semantic network is a matrix.

kind of microformat index (e.g., `vevent` index).

Resum: It is a data structure in matrix form. Its goal is to allow the user choosing a specific web page and a particular type of microformat.

Web page of microformats: Once a cell of Resum is selected we prepare a web page with the microformats found.

V. A PRACTICAL EXAMPLE

In this section we introduce a guided example an emphasize on the sense and goal of using a software agent like the *semantic analyzer*.

A typical session begins with a search of semantic relations. For this, we launch the *semantic relation searcher* as it is depicted in the Figure 7.

Then, the filtered URL's from de Google search are collected in order to prepare the sample for analysis, the user can choose the sample size. For this, the tool offers a page (in the tool's interface) which is called `Sample` where users can edit the list of URL's, see Figure 8.

Once we have a well defined sample, the next step is to click on the button `Sem. Analysis` of the tool's interface. The tool brings each web page and traverses the proper (X)HTML code searching microformats. The tool's interface has an area to report the number of microformats found in each visited web page (see Figure 9). Finally the user can click on the button `Sem. relations` in order to view a

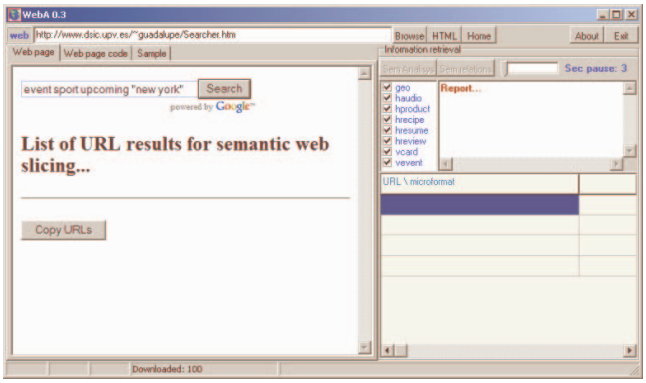

Figure 7. The searcher web page.

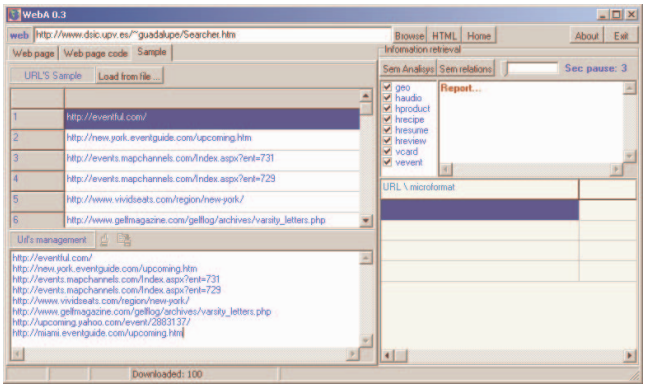

Figure 8. The tool's sample section.

more friendly presentation of the result report. If the user clicks on the presented resum, a list of microformat names corresponding to a web page and a kind of microformats is deployed. See Figure 10.

Despite of the usefulness of microformats, web developers do not use them profusely. Here, in the Table I we show different queries that were launched to Google by means of the web page searcher and the number of microformats discovered. That results were determined by the semantic relation searcher tool. A comparison with similar tools is not possible because they perform analysis only of isolated web pages. In such a case our tool offers a more detailed result enriching Google's results.

Table I
SEARCHING MICROFORMATS

Google search query	vcard mF	vevent mF	geo mF
event sport upcoming "new york"	0	15	0
personal service "Los Angeles" street	11	0	1
"medical services" Madrid hospital	20	0	0
song author	12	0	0

Figure 9. The downloading process and its running report.

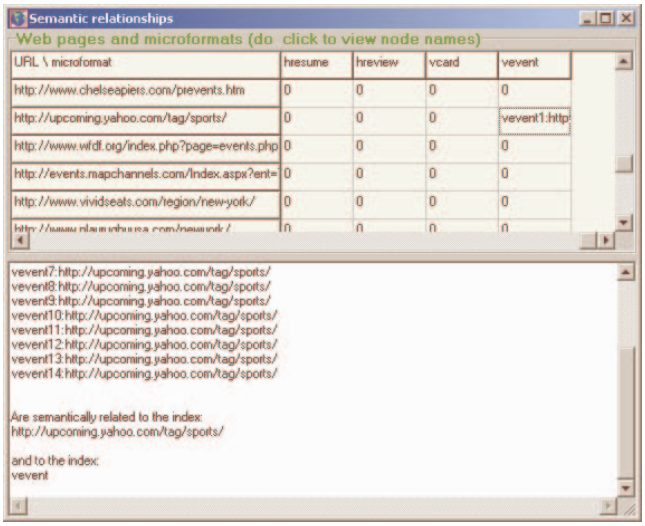

Figure 10. Resum of search result.

A. *The sense of the semantic relation search and semantic analyzer*

When a user performs a searching session, she writes a query in the search web page and waits for a list of results related to the introduced keywords. Let us consider a tourist visiting "New York". If she wishes to go to some sport events, she can check the newspaper or the web. If she chooses the web, then she must read many web pages and determine what are the preferred events. A modern approach, based on semantic web, could be the following: The user launches a software agent, makes a query to a web searcher, then asks to a semantic analyzer to extract the upcoming sport events in a set of web pages, and for each event, it would be added to an electronic appointment book. Thanks to the microformats this is possible.

The semantic analyzer software agent, requires a sample, in a similar way to the previous tool. The following URL list has the web page addresses that contain microformats related with *upcoming sport events in New York*.

```
http://eventful.com/
http://new.york.eventguide.com/upcoming.htm
http://www.chelseapiers.com/prevents.htm
http://upcoming.yahoo.com/tag/sports/
http://www.wfdf.org/index.php?page=events.php
http://events.mapchannels.com/Index.aspx?ent=731
http://www.vividseats.com/region/new-york/
http://www.playrugbyusa.com/newyork/
```

A difference w.r.t. the semantic relation searcher is the result, the semantic analyzer extracts a slice from the semantic network, i.e., a set of nodes, or better a set of microformat code (see Figure 11) while the previous one restores only the name and kind of the microformats.

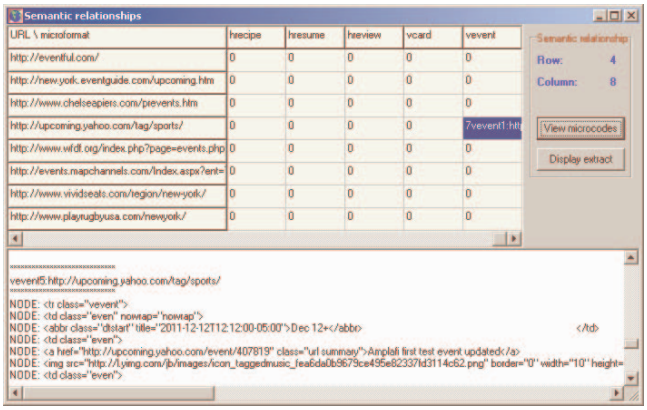

Figure 11. Resum of semantic analysis.

Now, if we wish to view the extracted microcodes we can click on the `Display extract` button, and a web page with the extracted microformats will be deployed, as it is shown in 12.

Figure 12. Extracted microformat in (X)HMTL view.

VI. Conclusion

In this work we have presented a semantic analyzer software agent that allows us to discover information units called microformats and also the tool extracts the code

associated to them. The software agent can be conceived as an auxiliar tool in order to find complete units of information in a set of pages and discard those web pages that do not embed that particular kind of information. The tool offers a report specifying where microformats are and their type, thus, the user amount of work can be reduced because the set of potentially useful web pages is smaller than the produced by a web search.

We do not claim for semantic equality nor semantic equivalence between similar class of microformats located in different web pages, we discover certain semantic relationship as it was defined in our setting. We think this is useful and sufficient in order to build a report with a map of meaningful units of information embedded in sets of web pages.

Microformats are a convenient method to represent in a pragmatic way the set of semantic web. We have referenced to [17], and [18] who have developed high quality tools for microformat viewing and web filtering. However they process isolated pages. Our approach could be considered not only as a procedure for microformat extraction, but also as an interesting way to improve the quality of web search results, and also as an interface to connect external and useful tools. For instance, for event registering, for people discovering, etc.

The future work is to connect the procedure presented in this paper with external tools, and also to develop a browser or add-ons that automatically report the relation between multiple pages.

REFERENCES

[1] WorldWideWebSize.com, "The size of the World Wide Web. Accessed on July 24th 2009." `http://www.worldwidewebsize.com/`.

[2] J. Bradshaw, *Software Agents*. MIT Press, 1997.

[3] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American Magazine*, May 2001.

[4] L. Yu, *Introduction to the Semantic Web and Semantic Web Services*. Chapman & Hall/CRC, 2007.

[5] T. Çelik, "What's the Next Big Thing on the Web? It May Be a Small, Simple Thing - Microformats," *Knowledge@Wharton*, 2005.

[6] Microformats.org, "The Official Microformats Site." `http://microformats.org/`, 2009.

[7] R. Khare and T. Çelik, "Microformats: a Pragmatic Path to the Semantic Web," in *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, pp. 865–866, ACM, 2006.

[8] hCalendar, "Simple, Open, Distributed Calendaring and Events Format." `http://microformats.org/wiki/hcalendar`, February 2009.

[9] R. Khare, "Microformats: The Next (Small) Thing on the Semantic Web?," *IEEE Internet Computing*, vol. 10, no. 1, pp. 68–75, 2006.

[10] J. G. Ramos, J.Silva, G. Arroyo, and J. Solorio, "Information retrieval from the semantic web based on microformats and semantic networks," in *Seventh International Andrei Ershov Memorial Conference: Perspectives of System Informatics*, 2009. To appear.

[11] J. F. Sowa, "Semantic Networks," in *Encyclopedia of Artificial Intelligence* (S. C. Shapiro, ed.), John Wiley & Sons, 1992.

[12] R. Quillian, "Semantic Memory," in *Semantic Information Processing* (M. Minsky, ed.), MIT Press, 1969.

[13] G. Stumme, B. Hoser, C. Schmitz, and H. Alani, eds., *ISWC 2005 Workshop on Semantic Network Analysis*, vol. 171 of *CEUR Workshop Proceedings*, (Galway, Ireland), 2005.

[14] H. Alani, B. Hoser, C. Schmitz, and G. Stumme, eds., *Proceedings of the 2nd Workshop on Semantic Network Analysis*, 2006.

[15] J. F. Sowa, ed., *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann, 1991.

[16] hCard, "Simple, Open, Distributed Format for Representing People, Companies, Organizations, and Places." `http://microformats.org/wiki/hcard`, 2009.

[17] C. Yu, "Tails add-on." Available at: `http://blog.codeeg.com/tails-firefox-extension-03/`, 2007.

[18] J. Silva, "Web filtering toolbar 1.3." Available at: `https://addons.mozilla.org/es-ES/firefox/addon/5823`, 2008.

[19] GoogleCode, "Google AJAX Search API." Available at: `http://code.google.com/intl/en/apis/ajaxsearch/`, 2009.