An Evolutionary Approach to the Semantic Web

Oren Etzioni University of Washington etzioni@cs.washington.edu
Steve Gribble University of Washington gribble@cs.washington.edu
Alon Halevy University of Washington alon@cs.washington.edu
Henry Levy University of Washington levy@cs.washington.edu
Luke McDowell University of Washington lucasm@cs.washington.edu

Proposals for creating a semantic web have been around at least since 1995 [Dobson and Burrill, 1995]. A wide range of semantic markup languages have been proposed including RDF, N3, SHOE, DAML, and OIL. Yet, in contrast with the explosive growth of HTML, the semantic web has been slow to materialize. Below, we attempt to explain why by articulating several hypotheses. We then sketch our own approach which seeks to facilitate gradual evolution from HTML to a semantic web.

People have not been tagging their web pages because they have had no reason to do so.[1] After all, HTML became popular only after MOSAIC came along. Tagging will be driven by applications that consume the tags and result in immediate, tangible satisfaction for the author; **Instant gratification** is a requirement for rapid adoption.

Another factor to consider is ease of authoring. HTML spread like wild fire due to its simplicity; users did not need documentation. Instead, cut-and-paste sufficed to create the average home page. To proliferate, the tagging scheme has to be **simple** to understand and convenient to author. Clean syntax is key; usability is essential. Moreover, **Self-documenting tags** whose approximate meaning is apparent from their name, and are documented by very brief text, will proliferate before tags whose semantics requires elaborate specification.

Unlike 1993, we now have upwards of a billion HTML pages. **Backward compatibility is essential:** HTML pages will be tagged without disrupting the standard HTML-based browsing and searching activities. In addition, whereas much of the research on the semantic web has focused on enabling SQL-style queries and powerful agents, we believe that tagging will impact browsing and searching before they support more "semantically intense" applications.

Finally, we believe in **Local tags.** That is, local communities will adopt tagging schemes long before these spread across multiple organizations. We are testing our tagging scheme (and our working hypotheses) by attempting to gradually evolve a miniature semantic web in the University of Washington Computer Science Department, complete with a simple set of tags, instant gratification "apps.", and authoring tools. Because of our emphasis on *local* tagging, we have deferred committing to a mechanism for "semantic interoperability" such as XML namespaces in RDF or the use-ontology tag in SHOE. However, we anticipate leveraging the ontology-matching mechanisms outlined in [Doan *et al.*, 2001].

Our tagging scheme was designed to embody the above hypotheses. We introduce it via a simple example in Figure 1. The tags are self-documenting, but two aspects are noteworthy. First, nested tags indicate attribute information. For example, <**office hours**> is an attribute of each <**instructor**>. Second, for convenience we enable taggers to tag an entire list or table by specifying a simple regular expression at the top. The "..." refers to the text to be tagged.

We believe that even modest amounts of tagging will enable useful applications. For instance, the simple annotations in our example would enable a semantically aware search engine to automatically extract the location for a course in response to a query. In our department, too much information is gathered and updated manually

```
<course>
<h1><name>Networking
Seminar</name></h1>
<p>All meetings held at
<time>1 p.m.</time> in
<location>Sieg 134</location>.
<b>Refreshments</b> will be served.
<p>Office hours for additional
assistance:
<instructor>Prof.  John Fitz
    <office hours>Tue 3-4 p.m.</office hours>
</instructor>
<instructor>Prof.  Helen Randolph
    <office hours>Fri 9-10 a.m.</office hours>
</instructor>.
<table> <tr><th>2002 Schedule
<reglist='<tr><td><date>...</date>
            <td><topic>...</topic></tr>'>

<tr> <td>Jan 11 <td>Packet loss</tr>
<tr> <td>Jan 18 <td>TCP theory</tr>
</reglist></table>
</course>
```

Figure 1: Example of tagged HTML.

(e.g., a departmental phone directory, Who's Who, and calendar). Semantic tags would enable automatic applications to replace these manual efforts. For example, the authors of home pages for courses, reading groups, and other departmental events could tag their pages and "publish" them to a database.[2] Then, a departmental calendar could be created automatically by simply querying that database. As a result, the page's author will have the instant gratification of including his event in the calendar (and potentially elsewhere) by merely tagging the event's home page.

Beyond supplanting manual information gathering, we believe that our distributed and automated approach will lead people to more ambitious information integration efforts (e.g., a department-wide publication database). The application-independent format of tags will enable new applications to exploit the semantic data in unanticipated ways.

Our approach is closest in spirit to the "lightweight databases" of [Dobson and Burrill, 1995], but we propose a different syntax, distinct applications, and a novel approach to facilitating adoption. Our measure of success for this initial phase of the project is the number of authors, pages, and tags that become part of our effort in the coming months.

References

[Doan *et al.*, 2001] A. Doan, P. Domingos, and A. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine Learning Approach. In *Proceedings of the ACM SIGMOD Conference*, 2001.

[Dobson and Burrill, 1995] S. A. Dobson and V. A. Burrill. Lightweight databases. *Computer Networks and ISDN Systems*, 27(6):1009–1015, 1995.

[1] For brevity, we refer to the act of authoring semantic web content as *tagging*.

[2] Publication is achieved by submitting the tagged page's URL to a servlet that reads the tags and issues appropriate database updates.