

# Semantic Advertising for Web 3.0

Edward Thomas, Jeff Z. Pan, Stuart Taylor, Yuan Ren, Nophadol Jekjantuk,  
and Yuting Zhao

Department of Computer Science University of Aberdeen  
Aberdeen, Scotland

**Abstract.** Advertising on the World Wide Web is based around automatically matching web pages with appropriate advertisements, in the form of banner ads, interactive adverts, or text links. Traditionally this has been done by manual classification of pages, or more recently using information retrieval techniques to find the most important keywords from the page, and match these to keywords being used by adverts. In this paper, we propose a new model for online advertising, based around lightweight embedded semantics. This will improve the relevancy of adverts on the World Wide Web and help to kick-start the use of RDFa as a mechanism for adding lightweight semantic attributes to the Web. Furthermore, we propose a system architecture for the proposed new model, based on our scalable ontology reasoning infrastructure TrOWL.

## 1 Introduction

Advertising is the main economic force which drives the development of the current and future World Wide Web. According to a report by PriceWaterHouseCoopers [4] advertising revenues totalled \$6.1 billion for the fourth quarter of 2008, an increase over the previous year even during an economic recession. Of this, banner advertising accounted for the second largest piece of this revenue, following search revenue with 21 percent of the total market.

Search revenue typically uses the keywords entered by the user to match against keywords which have been purchased by an advertiser. This is a strict match - advertisers who wish to cover synonyms or hyponyms of a particular keyword will purchase additional keywords. Since advertisers only pay per impression, or per click, there is no penalty to covering wide ranges of keywords. The simple matching of keywords entered by a user, and keywords purchased by an advertiser makes it easy to understand, and hence a popular route for advertisers.

Matching banner adverts to web pages is a harder problem. In this case, the entire content of the web site, and the context of the web site which holds it must be taken into account. Some systems, such as Google AdSense, attempt to extract the most important (by some information retrieval metric) keywords from the page, and match these to keywords selected by an advertiser [5]. There is a very low cost of entry to systems like this, and publishing or advertising on these networks is a trivial pay-as-you go process. Other systems, such as those

used by DoubleClick<sup>1</sup>, work closely with a publisher to classify their web site according to a taxonomy of content, and may embed custom tags or keywords into the page itself to improve the matching process. There is a large cost of entry to advertising or publishing in this way, DoubleClick and similar networks only take on web sites with a certain minimum number of ad impressions per month. The relationship among publishers, advertising agencies, and advertisers, is much closer than the relationships found in traditional media with a large degree of human involvement in the design and deployment of advertising campaigns. This relationship is costly in terms of man-hours, and requires a large number of ad impressions to make it viable.

This paper outlines a third alternative for Web 3.0. By using lightweight semantics on a web page, and RDF descriptions of adverts (and more importantly, what web sites should a particular advert appear on), combined with some existing semantic web technologies, we can produce an open market for online advertising which offers automatic targeting, with more accurate targeting, combined with the zero cost of entry which keyword based advertising currently operates.

In this paper, we will first discuss the technical motivations of our approach, before proposing a new model for online advertising, based on lightweight embedded semantics. Furthermore, we propose a system architecture for the new model, based on our scalable semantic reasoning infrastructure TrOWL [10], which provides scalable query answering over a wide range of ontology languages from RDF to OWL. Finally we will present two case studies on Semantic Advertising, and conclude the paper with a discussion of areas of future work.

## 2 Approach

Traditional approaches such as strict keyword matching are quite limited in a sense that it can not disambiguate the keywords in different context. Also, the synonyms or hyponyms have to be manually specified by the advert providers but not automatically derived. Other approaches such as the one from DoubleClick require large amount of work for both the advert provider and advertising agency to classify the web site's content and fit it into a pre-defined taxonomy. This is inconvenient for own of small web sites. Furthermore, when the web page is automatically generated in real-time it is difficult to apply such an approach.

Our approach attempts to provide a more accurate and easy-to-use matching between web pages and adverts by making use of semantics embedded in both. This can, on the one hand, enable the web developers and advert providers describing their documents (web pages and adverts) and requirements in a intuitive and flexible manner, and on the other hand, make use of existing semantic web resources such as ontologies, thesaurus and reasoners to discover the relations in between. The advert providers no longer need to worry about issues such as the synonyms because they will be inferred automatically with the help of upper level categorisation ontologies; while the web owners no longer need to classify their web pages one by one because the embedded semantics tells everything.

---

<sup>1</sup> DoubleClick: <http://www.doubleclick.com/>

This approach includes two major aspects: (1) the automatic reasoning in matching and (2) the manual or automatic annotation of the documents. Like any other web-based application, a crucial technical feature of this service is efficiency. Neither the web publisher, nor the advertising provider would like to an advert matching delays the rendering of the web page. In the semantic web context, the efficiency of a reasoning-related service is strongly restricted by the language used to describe the semantics. Currently, the de facto semantic web languages recommended by W3C are RDF, RDF Schema, OWL and their dialects. OWL family are based on well-defined and understood description logics (DLs) with many mature tool supporting. However, many OWL dialects, such as OWL DL and OWL2 DL, are expensive in reasoning. RDF, on the other hand, is widely applied in web data exchange and integration; however, they have limited expressive power. One solution is to use TrOWL, which provide scalable reasoning and query answering services for not only RDF-DL and OWL2 tractable profiles<sup>2</sup>, including OWL2-QL, OWL2-EL and OWL2-RL), but also expressive ontology languages such as OWL DL and OWL2-DL. The main idea here is that user ontologies in OWL DL and OWL2-DL can be approximated to tractable profiles by TrOWL (based on quality guaranteed approximation-based reasoning [8, 9]). For query answer, the target profile is OWL2-QL, which uses SQL engine for query answering after semantic query writing based on semantic approximation. OWL2-QL is a super-language of RDF-DL, hence the approach works well with RDF-DL ontologies. For classification and subsumption checking, the target profile is OWL2-EL.

As for the annotation aspect, one can use vocabulary defined in ontologies to annotate web page. One way is to create an RDF document and render it as in HTML through transformation techniques such as XSLT. For web developers, it will be more convenient to embed RDF annotation into normal web pages and further validate them w.r.t. its schema. RDFa, an application of RDF bridges the gap between web page composing language such as XHTML and RDF. It can express structured data such as RDF in any markup language by specifying attributes of web page elements. In this paper, we use RDFa to annotate the documents and to enhance them with lightweight semantics in RDF.

### 3 System Architecture for Semantic Advertising

In this section, we propose the system architecture for semantic advertising. Firstly, we will describe the role of the content publisher. Secondly, we are going to explain the role of Advertising provider. Then, we will clarify how the system fits together. Figure 1 shows a proposed system architecture for semantic advertising.

#### 3.1 Web site owner/web developer

The content publisher creates the web page with embedded the semantic web data. At this step, they may need some tools to help them decorate the XHTML

<sup>2</sup> <http://www.w3.org/2007/OWL/wiki/Profiles>

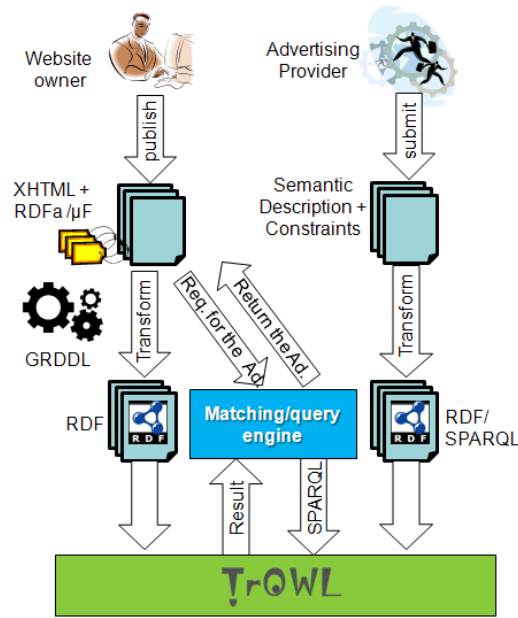


Fig. 1. System Architecture for a Semantic Advertising

page with RDFa [1] or Microformats [6], these are currently new formats but support is being included in many tools and content management systems (for an example, see <http://www.cmswire.com/cms/web-cms/rdfa-drupal-and-a-practical-semantic-web-004149.php>). Then, subscribe their web site to the semantic advertisement system. The publisher is then given a code snippet to include on all pages, at the position where the advert should appear. This process is identical to current keyword based approaches. The first time any unique page requests an advert, the advertisement system will retrieve the page and extract the embedded semantics as RDF. This RDF graph is then cached in a repository and used to match suitable adverts.

### 3.2 Advertising provider

The advertising provider publishes a description of each advert they wish to run. This contains a technical description of the advert, including its format (text, image, video, flash animation, or interactive), its size, and any particular layout requirements it imposes. The description also contains one or more sets of constraints on what type of content the advert should appear on, and give a schedule of how much the advertiser is willing to pay to display the advert on a page which fulfills each set of constraints. By doing this, it is possible for an advertiser to offer a more lucrative contract when it is advertising on content

which is more likely to bring customers, but still get broad exposure for a lower cost. We envision a web application with some functionality to generate these constraint sets for the advertiser.

### 3.3 The Advertising Broker

The advertising broker provides a repository for storing the descriptions of web pages, and the descriptions of advertisements and the constraints of the advertisers. In addition to this, it is important that the advertisers have access to background knowledge which they can use in their constraints to improve the matching possibilities. We will examine this more fully in the MusicMash case study.

Since background knowledge can be in any format, it is important that the broker allows RDF, RDFS, and OWL information to be stored and queried in a sound manner. The TrOWL system uses techniques such as quality-guaranteed approximation [8, 9] to reduce querying across all these formats to query answering across OWL2-QL [3] and OWL2-EL [2].

Finally, the advertising broker must also perform the tasks associated with any advertising delivery: ensuring that adverts are rotated and not allowed to get stale; selecting the advert which offers the best revenue stream for the publisher; and performing the basics of hosting, billing, etc.

### 3.4 How does system work?

The semantics embedded on the page will be converted into RDF graphs, and the constraints given by the advertisers will be rewritten as SPARQL queries. By running each query against the repository of graphs extracted from content, we can produce a map of the best advertising for each web page. We propose that the advertising system performs the matching process at the point when new content or new adverts are added to the system. This can then be stored in a cache to improve performance on repeated matching.

When a user requests an advert for a particular page, the system can consult the map of appropriate adverts and select the most lucrative. Additional techniques could, for example, ensure that a user does not see the same advert on the same site too many times, but this is outside the scope of this paper.

## 4 Case Studies

### 4.1 Product Blog

This web site considers a bog style web site which publishes news and reviews of mobile phones, computers, laptops, and other electronic products. Web Sites which fall into this broad category include Gizmodo<sup>3</sup>, Stuff<sup>4</sup>, and Pocket Lint<sup>5</sup>.

---

<sup>3</sup> <http://www.gizmodo.com>

<sup>4</sup> <http://stuff.tv>

<sup>5</sup> <http://www.pocket-lint.com>

These web sites are extremely important to manufacturers as they provide a key line of communication to early adopters of new technologies, and an informal review of the homepages of these three web sites shows that the advertisers are all for the types of products which are likely to be reviewed or featured on the site.

We will outline how a similar web site might make use of semantic technologies to make it easier to match suitable adverts to particular articles. In this case study, we will use as an example a review of a digital camera, taken from Pocket Lint, from <http://www.pocket-lint.com/reviews/review.phtml/3526/nikon-dslr-D90-dslr-camera.phtml>. By looking at the metadata that can be gleaned from the review, we will first consider the RDFa annotations which may be chosen:

```
PREFIX rev: <http://purl.org/stuff/rev#>
PREFIX dc: <http://purl.org/dc/elements/1.1/date>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX shop: <http://amazon.com/NS/products#>

<http://pocket-lint> dc:publishes <> .
<http://www.nikon.com/products/D90> rev:hasReview <> .
<> a rev:Review;
  dc:subject <http://www.nikon.com/products/D90>;
  rev:title "Nikon D90 DSLR camera";
  rev:reviewer <http://www.pocket-lint.com/author.phtml/41>;
  rev:rating "9"^^xsd:decimal;
  rev:text "Nikons DSLR boffins have been...";
  shop:price "649.99"^^xsd:decimal;
  dc:date "2009-10-10"^^xsd:date;
  skos:related <http://www.pocket-lint.com/reviews/olympus-e-520>,
    <http://www.pocket-lint.com/reviews/olympus-e3>,
    <http://www.pocket-lint.com/reviews/canon-eos-50d>,
    <http://www.pocket-lint.com/reviews/nikon-d3x>,
    <http://www.pocket-lint.com/reviews/olympus-e-30> .
```

This fragment of RDF describes in semantic terms, the main content of the review. It uses three existing and well used vocabularies so that the semantics of the properties used will be well known. This RDF first states that the current page (denoted with <>) is a review, and that it is a review on a particular thing which is the subject of the URI given - this URI can be dereferenced to find it is a Nikon D90 digital SLR - we may also find more RDFa embedded on the dereferenced page which will describe this camera in more detail. Simple metadata on the review then follows, giving the title and date of the review, the reviewer, and the rating of the review, as well as the full text of the review in this case we have truncated the text for reasons of space, but the full article could be annotated with this property with only a trivial addition of the appropriate RDFa tag. Finally, the article offers some related reviews which we link to.

As we have previously stressed, all of this information is currently present in the article, but it is not in a format that can be clearly understood by software.

Now we will consider the approach of two potential advertisers.

**Electronic Store** The first is a discount electronic store (called EStore, having the URI <http://estore.com>). which sells cameras, and the second is a camera manufacturer which sells a competing product. The camera store enumerates the constraints on where it wishes to place its advertising as:

- The advert should only appear on reviews of the same product
- The review in question must be a favorable review
- The price that the store sells the product at should be at least 10% less than the price quoted in the review

The advertiser must then use RDFa to describe the products he lists on his site, and submit this information to the RDFS repository which manages the advertising. The constraints on the advertising are then encoded as a SPARQL query:

```
PREFIX rev: <http://purl.org/stuff/rev#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX shop: <http://amazon.com/NS/products#>

SELECT ?x ?advert WHERE {
  ?x a rev:Review;
     shop:price ?price;
     rev:hasRating ?rating;
     dc:subject ?product .
  ?advert dc:subject ?product;
          ad:target ?page.
  ?page shop:price ?ourprice;
         dc:publisher <http://estore.com> .
  FILTER (?theirprice > (?ourprice * 1.1))
  FILTER (?rating > 7)
}
```

This query will return tuples containing a map between the adverts being published by EStore and the Web pages which contain reviews which are suitable for each advertisement, within the bounds of the constraints they have set. Further constraints may give a basis for the contract between the advertiser and publisher. The publisher requires the best return for each click on the advertisement, so the results are ordered by the amount the advertiser is willing to pay.

When a user visits the review of the Nikon D90, the advert is requested and the URI of the requesting page is included in the HTTP request as the Referrer field. The advertising system will be lookup the RDF which describes the page and find all adverts whose constraints match the RDF, using a simple set of heuristics the most lucrative advert can be found and returned to be displayed on the page. When the advert is requested from the advertiser, they too can

lookup the RDF describing the page on which it is to be hosted, either on the server side, or for rich media adverts, on the client side, to further customise the advert to the page on which it will reside. In this case, the advert could highlight their exact price advantage over the recommended retail price.

**Competing Manufacture** Here we imagine the requirements of Canon, who compete with Nikon who make the D90 featured in the review. In their case, they may wish to advertise only when they know that they have competing product, and where that product has a better review on the same web site. Their requirements are:

- Only advertise on products which compete with ours
- Only advertise where the same web site carries a review of the competing product
- Only advertise where our product has a better review than the competing product

To formalise these requirements, we can use the SKOS property “related” to find related pages which are also reviews.

```
PREFIX rev: <http://purl.org/stuff/rev#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX shop: <http://amazon.com/NS/products#>
```

```
SELECT ?x ?advert WHERE {
  ?x a rev:Review;
     rev:hasRating ?competitorRating;
     dc:related ?related.
  ?related a rev:Review;
     dc:subject ?relatedproduct;
     rev:hasRating ?ourRating.
  ?advert dc:subject ?relatedproduct;
  ?relatedproduct shop:madeBy <http://canon.com> .
  FILTER(?ourRating > ?competitorRating)
}
```

This simple query matches whenever a review lists one of Canon’s products as a related product, and when the related product has a better review score than the competing product. The advert could point out this fact when it is generated, it could include the two scores, or it could contain any other information from the RDF which was useful.

## 4.2 MusicMash2

This case study examines the MusicMash2 web site<sup>6</sup> which delivers music videos for users based on a semantic mashup of ontologies and folksonomies [7]. The

<sup>6</sup> <http://www.musicmash.org>

web site currently displays keyword based adverts using Google AdSense, but due to the ambiguous nature of some band names, the resulting adverts are often inappropriate. The web site embeds RDFa descriptions of the video content on every page. It uses a MusicBrainz URI to uniquely identify the track. A sample of the embedded RDF is shown below:

```
PREFIX video: <http://purl.org/media/video#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mb: <http://musicbrainz.org/track/>

<#video> a video:Recording;
  dc:title "Hotel California";
  dc:subject mb:d548a707-e0a8-48a0-94b4-e267793a918e.
```

The advertiser in this case has a music store and wishes to place adverts containing lists of CDs which containing the track. To do this, the advertiser needs to supply some additional background knowledge in its constraints.

To support this, the advertiser uploads the Musicbrainz RDF export to the broker. This contains every published record, along with a list of tracks, and links to the artist who performed them. Furthermore, the music store must provide a list of the albums they currently sell. Since this is may change frequently, they can offer this as a SPARQL query, packaged into a URI, which can be included in the query:

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX mb: <http://musicbrainz.org/>
PREFIX shop: <http://amazon.com/NS/products#>

SELECT ?x ?album ?advert
FROM <http://trowl.eu/default>
FROM <http://musicstore.com/sparql?query=CONSTRUCT...>
WHERE {
  ?x dc:subject ?track.
  ?track mb:appearsOn ?album.
  <http://musicstore.com> shop:sells ?album.
}
```

This query differs from the previous queries in that it explicitly imports an external dataset. This means that the query engine will contact the SPARQL server operated by the music store to retrieve a list of albums it sells, and can then filter it to only return pages which contain albums they sell. Because the query is specifying an external graph, it must also explicitly include the default repository (which in TrOWL always has the same URI).

This therefore includes two distinct external knowledge sources. The background knowledge which is imported from MusicBrainz allows the basic information present in the semantic markup to be expanded to also derive album information, and the dynamic inclusion of another SPARQL endpoint further allows the query to include up to the minute information on exactly what albums are currently in stock and for sale.

## 5 Conclusions and Future Work

In this paper we have outlined a vision for publishing advertising specifications and matching these to semantically enabled web pages. We see this as a general approach that can work across a number of different domains without changing the underlying method. There are some issues still to resolve before this can be realised on a large scale.

The first and most difficult problem is that embedded semantics are currently not widely used on commercial web sites. RDFa is a new format which is not greatly understood, and also there is no compelling application for these semantics which would encourage large publishers to add them to their web sites. Our hope is that by giving a financial incentive for web sites to deploy RDFa, by improving the matching of advertisements to web pages, we may help to bootstrap these new technologies into the mainstream.

The second issue occurs on highly dynamic web pages, where the content is different for every user. The cost of performing the extraction of RDFa, RDFS reasoning, and matching this to the most suitable advert would make this method prohibitive for these web sites. There is some research being made into methods for approximate matching and querying.

## References

1. RDFa in XHTML: syntax and processing, W3C recommendation, 2008.
2. Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. Is tractable reasoning in extensions of the description logic  $\mathcal{EL}$  useful in practice? In *Journal of Logic, Language and Information, Special Issue on Method for Modality (M4M)*, 2007.
3. D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, and G. Vetere. DL-Lite: Practical Reasoning for Rich DLs. In *Proc. of the DL2004 Workshop*, 2004.
4. Price Waterhouse Coopers. IAB Internet Advertising Revenue Report, 2008.
5. Harold Davis. *Google Advertising Tools: Cashing in with AdSense, Adwords, and the Google APIs*. O'Reilly Media, Inc., 2006.
6. Rohit Khare. Microformats: The next (small) thing on the semantic web? *IEEE Internet Computing*, 10(1):68–75, 2006.
7. Jeff Z. Pan, Stuart Taylor, and Edward Thomas. MusicMash2: Mashing Linked Music Data via An OWL DL Web Ontology. In *Proceedings of the WebSci'09: Society On-Line*, 2009.
8. Jeff Z. Pan and Edward Thomas. Approximating OWL-DL Ontologies. In *the Proc. of the 22nd National Conference on Artificial Intelligence (AAAI-07)*, pages 1434–1439, 2007.
9. Yuan Ren, Jeff Z. Pan, and Yuting Zhao. Soundness Preserving Approximation for TBox Reasoning in Description Logic R. In *Proc. of the 22nd International Workshop on Description Logics (DL2009)*, 2009.
10. Edward Thomas, Jeff Z. Pan, and Yuting Zhao. TrOWL: A Scalable Ontology Reasoning Infrastructure for OWL2. In *Proceedings of the European Semantic Web Conference*, 2009.