EFFICIENT SEMANTIC SEARCH USING MICROFORMATS

Thesis is submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Software Engineering

Submitted By
Prince Bhanwra
(801031024)

Under the supervision of:
Dr. Seema Bawa
Professor (CSED), Dean (Student Affairs)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*Efficient Semantic Search using Microformats*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Seema Bawa* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(**Prince Bhanwra**)
(801031024)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

04/JUNE/2012

(**Dr. Seema Bawa**)
Professor, Dean (Student Affairs)
Computer Science and Engineering Department,
Thapar University,
Patiala

Countersigned by

12/6/12

(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

It is a great pleasure for me to acknowledge the guidance, assistance and help I have received from **Dr. Seema Bawa**, Professor (Computer Science and Engineering Department), Dean (Student Affairs). I am thankful for her continual support, encouragement and invaluable suggestions. I was provided with not only help but resources required to complete my thesis on time.

With great pleasure and acknowledgement, I extend my profound thanks to **Dr. Maninder Singh**, Head (Computer Science and Engineering Department) for his cooperation and help.

I would also like to thank all the staff members of Computer Science and Engineering Department for providing me all the facilities required for the completion of my thesis work.

I am highly thankful to my parents for the inspiration and ever encouraging moral support which enabled me to pursue my studies.

Prince Bhanwra

Abstract

Increasing information on internet has made difficult for users to find relevant information on Web. Querying search engine returns thousands of results, most of them irrelevant. The core problem is broken, unlinked web. There is a need for structured web with semantic search engines understanding the information and resources on it; returning few but relevant results. RDF/XML, OWL and Microformats are the popular new technologies in use for describing information and resources on the web. Out of these microformats is chosen in this thesis for efficient semantic search. A website is specially designed as a part of thesis work embedded with microformats semantic markup. A search engine is also created to query this website. Efficiency of the semantic search engine is being verified via test cases designed for this purpose; which are made to run on our microformats based search engine and regular search engines such as Google and/or Yahoo/Bing etc.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Ever since the World Wide Web blossomed into the general computing in the mid-1990s; all efforts have been made to extend its basic presentation richer and more meaningful creating a network of information. We have now envisioned a web with the information that can not only be read by humans but also be understood by computers. The web could evolve from a collection of loosely linked pages to an enormous database that could be searched and filtered and re-assembled in new ways. The Semantic Web introduces numerous new technologies for describing information and resources on the web such as RDF/XML, OWL and Microformats. The Resource Description Framework (RDF) is a language for representing metadata [**1**] [**2**] about Web resources (such as the title, author etc.) or the availability schedule for some shared resource in the World Wide Web. OWL is designed to provide a common way to process the content of web information (instead of displaying it). It is designed to be read by computer applications called hidden web which is machine understandable and is not visible to users. OWL and RDF are much of the same thing, but OWL is a stronger language with greater machine interpretability than RDF [**3**]. Whereas microformats are designed for humans first and machines second.

Microformats are set of simple, open data formats built upon existing and widely adopted standards. Instead of throwing away what works today, microformats intend to solve simpler problems first by adapting to current behaviours and usage patterns (e.g. XHTML, blogging) [**4**].

It works on the basis of XHTML [**5**], which together with CSS has revolutionized the web, creating webpages with looks and behaviour similar among different browsers on different operating systems. Just by adding a Microformats class attributes do the work right, making data machine understandable. It is a new way of thinking about data, following simple open data format standards, actively developing and implementing for more/better structured blogging and web microcontent publishing in general [**4**].

1.1 Why Microformats?

The primary advantage of Microformats over RDF is that we embed metadata directly in the XHTML called Visible Web, thus reducing the amount of mark-up we need to write i.e. we need not write additional RDF apart from XHTML content. With Microformats instead of doing extra work adding semantics to the web, it is really simple. We need to be concerned more about creating our webpage instead of doing extra work to add semantics. Just a small extra step of adding little microformat mark-up to the content is enough and to perform task, various tools are available to produce mark-up in Microformats [**6**] and to store content on the remote servers.

The reason why microformats should be adapted is that it works on today's web and data in XHTML visible to users and would always result in accurate and updated information, the reason why metadata failed [1]. Microformats allows semantic information to be added to the web in such a way that it could be read, edited, viewed and verified by humans and also make sure it is machine understandable.

In our Microformats based Semantic search engine, we are going to exploit it's this capability to provide efficient and accurate search results. To perform this job, a website is specially designed to illustrate the power of microformats in search engine. In this website different types of microformats are being embedded in the website such as microformats showing personal information, Resume, Events, Views, House/organization address, Products, Bulletins, Advertisement, Media etc. and to query above said information, a search engine is embedded to the website. Google CSE XML API's are being used to power the search engine and to illustrate the real power of Microformats, search same keywords are being used in different context on the different pages other than intended, thus giving search engine a real environment where an ordinary pattern based search engine would fail whereas search engine working on the basis of microformats would succeed. Finally efficiency of the search engine is being checked via specially designed test cases. Test cases are made to run both the

[1]Issues with metadata are discussed on page 14.

microformats based search engine and normal search engines such as Google and/or Yahoo/Bing etc.

1.2 Semantic Web

The Semantic Web is a mesh of information linked up in such a way as to be easily process able by machines on a global scale. It is an efficient way of representing data on the World Wide Web (WWW) acting as a globally linked database.

World Wide Web consortium (W3C) looking towards the semantic web, working to improve, extend and standardize it. Many languages, publications, tools and so on have already been developed to support the semantic web. However Semantic Web technologies are still very immature. Little consensus about the likely direction and characteristics of the early Semantic Web is required. Majority of data on the web is in the broken form; not properly linked so it's difficult to use it on a large scale.

The Semantic Web is generally built on syntaxes which use URIs to represent data, usually in triples based structures: i.e. many triples of URI data that can be held in databases, or interchanged on the World Wide Web using a set of particular syntaxes developed especially for the task.

1.2.1 Need for Semantic Search?

The web is a gigantic network full of information. We use our favourite search engine to search for the things we need. Say "Punjabi food in London", it returns a bunch of links. We visit the website presented by the link, the website is actually a webpage. The browser interprets HTML and presents a nice clean page. No matter which browser we use, it presents the same page in a similar fashion. This is because HTML is the standard way of publishing pages on the web. We could imagine if we had different ways of publishing pages on the web. Then each browser would have to be programmed in a way to know which language the page was written. This would be such a pain.

However, HTML is not enough. When we search on the web, we are not interested in say a page about Punjabi food in Banglore; we are interested in finding information

about restaurants that serve Punjabi food in Banglore (restaurant names, location, price range, etc.). The issue is that the information we are looking for is usually on plain HTML page. What if we could have this information as structured raw data on the web so it could be easier to consume by other application instead of scraping an HTML page.

1.3 Semantic Technologies

RDF, OWL, Microformats and Microdata are the technologies introduced describing information and resources on web.

1.3.1 Resource Description Framework (RDF)

A graph is a representation of objects that are connected by links. In other words, we can have two things which are related in some way through a link that connects them.

Example:
Delhi is the capital of India. The two things in this sentence are Delhi and India. These two things are related by the link "is the capital of." And there we go: that is RDF! We could say almost everything we say in an English sentence can be represented in a graph form.

RDF consists of a triple:

- Subject – Subject here is "Delhi"
- Predicate – It's the link. Here, "is the capital of"
- Object – Object the other thing. Here, "India"

If we consider the relational model, you could have a table of cities where each row has information about a city, the table's attribute would include name, capital of, and a row would consist of "Delhi" underneath the name attribute and "India" under the capital of attribute. Essentially, RDF is just another way of representing data.

Example of an event written in RDF:

```
<rdf:RDF
xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
```

```
xmlns='http://www.w3.org/2002/12/cal/ical#'>
<Vcalendar>
<prodid>-//kanzaki.com//RDFCal 1.0//EN
</prodid>
<version>2.0</version>
<method>PUBLISH</method>
<component>
<Vevent>
<dtstart rdf:parseType='Resource'>
<dateTime>2012-12-30T00:00:00Z
</dateTime>
</dtstart>
<dtend rdf:parseType='Resource'>
<dateTime>2013-12-30T00:00:00Z
</dateTime>
</dtend>
<summary>Microformats: What the Hell Are They and Why Should I Care?</summary>
<description>Ryan King will explain why microformats are important and how you can mark-up specific kinds of content in ways that make it easier for
the right people to find your stuff.
</description>
<location>Balder Room</location>
<dtstamp>20051012T061505Z</dtstamp>
<uid>1129097705622@kanzaki.com</uid>
</Vevent>
</component>
</Vcalendar>
</rdf:RDF>
```

This piece of RDF basically says that this article has the title "The Semantic Web: An Introduction" and is written by "Sean B. Palmer".

1.3.2 Web Ontology Language (OWL)

The OWL Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans [**7**]. It is built on top of RDF, designed to be interpreted by computers and not designed for being read by people [**3**]. Ontology is about the exact description of web information and relationships between web information. It provides a common way to process the content of web information instead of displaying it.

OWL has three sublanguages and is written in XML:

- OWL Lite
- OWL DL (includes OWL Lite)
- OWL Full (includes OWL DL)

1.3.2.1 OWL Lite

OWL Lite was originally intended to support those users primarily needing a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. It was hoped that it would be simpler to provide tool support for OWL Lite than its more expressive relatives, allowing quick migration path for systems utilizing thesauri and other taxonomies. In practice, however, most of the expressiveness constraints placed on OWL Lite amount to little more than syntactic inconveniences: most of the constructs available in OWL DL can be built using complex combinations of OWL Lite features. Development of OWL Lite tools has thus proven almost as difficult as development of tools for OWL DL, and OWL Lite is not widely used.

1.3.2.2 OWL DL (includes OWL Lite)

OWL DL was designed to provide the maximum expressiveness possible while retaining computational completeness (either ϕ or $\neg\phi$ belong), decidability (there is an effective procedure to determine whether ϕ is derivable or not), and the availability of practical reasoning algorithms. OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (for example, number restrictions may not be placed upon properties which are declared to be transitive). OWL DL is so named due to its correspondence with description logic, a field of research that has studied the logics that form the formal foundation of OWL.

1.3.2.3 OWL Full (includes OWL DL)

OWL Full is based on a different semantics from OWL Lite or OWL DL, and was designed to preserve some compatibility with RDF Schema. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right; this is not permitted in OWL DL. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary. It is unlikely that any reasoning software will be able to support complete reasoning for OWL Full.

1.3.3 OWL is Different from RDF

OWL and RDF are much of the same thing, but OWL is a stronger language with greater machine interpretability than RDF. OWL comes with a larger vocabulary and stronger syntax than RDF.

It is built on XML's ability to define customized tagging schemes and RDF's flexible approach to represent data. The first level RDF required for the Semantic Web is an ontology language what can formally describe the meaning of terminology used in Web documents. If machines are expected to perform useful reasoning tasks on these documents, the language must go beyond the basic semantics of RDF Schema. The OWL Use Cases and Requirements Document provides more details on ontologies, motivates the need for a Web Ontology Language in terms of six use cases, and formulates design goals, requirements and objectives for OWL.

1.4 Microformats

Microformats is a web-based approach to semantic mark-up which aims to re-use existing HTML/XHTML tags to add semantics to web. This allows adding visible data to the web yet adding semantics to it, making it machine readable. The content of web pages is technically already capable of "automated processing", however such processing is very difficult because the traditional mark-up tags used to display information on the web do not describe what the information means. Microformats bridge this gap by attaching semantics and thereby obviate other, more complicated, methods of automated processing, such as natural language processing or screen scraping. The use, adoption and processing of microformats enables data items to be indexed, searched for, saved or cross-referenced, so that information can be reused or combined.

It allows information intended for end-users to be processed by machines automatically. Microformats are small patterns of HTML to represent commonly published things like people, events, blog posts, reviews and tags in web pages [**8**].It enables the publishing of higher fidelity information on the Web; the fastest and simplest way to provide feeds and APIs for the information in your website. They are designed for humans first and

machines second and are simple open data formats built upon existing and widely adopted standards. Instead of throwing away what works today, microformats intend to solve simpler problems first by adapting to current behaviours and usage patterns (e.g. XHTML, blogging) [**4**].

They tend to solve specific problems and aims to start as simple as possible. It believes in reusing building blocks from widely adopted standards and enable, encourage decentralized development, content and services. It is a new way of thinking about data, adapted to current behaviours and usage patterns highly correlated with semantic XHTML or lowercase semantic web [**4**].

Microformats is not a new language which is not an infinitely extensible and open ended. It is not an attempt to throw away how world works and enforce entirely new behaviours and patterns. It is not a panacea for all taxonomies, ontologies and other such abstractions but an intermediate between what web is today and what we dream it to be.

1.5 Microdata

Microdata is an extension to HTML5 that provides another way to embed microformats and poshformats [**9**] vocabularies. Design and simplicity of microdata is based on microformats principles; deliberately designed to help guide and create simpler, more usable and accessible solutions [**10**].

Microdata consist of a set of attribute extensions to HTML5:

- **itemprop** attribute is a more specific version of class.

- Properties that are not descendants of the element with the itemscope attribute can be associated with the item using the **itemref** attribute [**11**].

- **content** attribute on the meta element can be used to include invisible data that is not part of the content. As current browsers move meta inside <head>, make

sure to include via itemref. Conceptually similar to the 'value-title' feature of the value-class-pattern [**10**].

- **itemscope** attribute is used to create an item. It identifies blocks to be marked as structured data.

- **itemtype** attribute to specify the type for an item (for example: itemtype="http://microformats.org/profile/hcard")

Let's see the similarity between the microformats with microdata with the help of an example:

Microformats example:

```
<p class="vcard">
 Hi!! This is <span class="fn">Prince Bhanwra</span>. Catch me at  <a class="url" href="example.com">example.com</a>.
</p>
```

Microdata example:

```
<p itemscope itemtype="vcard" id="me">
Hi!! This is<span itemprop="name">Prince Bhanwra</span>.
<a itemprop="url" subject="me"  href="example.com">example.com</a>
</p>
```

In the above snippets we could notice the similarities between the microformats and microdata showing a vcard with sub elements illustrating name and URL.

Here root element "class" attribute has been replaced by "itemscope itemtype", where itemscope illustrates that it is a root element "itemtype" shows it's type; in our case vcard and the sub elements are replaced with "itemprop" i.e. item property and "fn" which stands for "formal name" has been replaced by "name" and class="url" is replaced by "itemprop" as illustrated above.

Chapter 2
Literature Review

Websites are designed with HTML, with the increase in internet users and data on Internet has made difficult for users to find relevant data on Web. Various search engines are used to find information on web. Search engines are designed to display search results called SERPS (search engine results page) widely on the basis of pattern matching. Search engines perform several actions to deliver search results such as crawling the web, making index, processing over web, calculating relevant object, and retrieving and displaying the information on the basis of search query. Pattern based search ideally gives thousands of search results; mostly irrelevant. Thousands of irrelevant results are retrieved because when you put in the search term, it matches those keywords with those available on webpages without finding any semantics relation between them. Some such queries which inherently ambiguity includes “java”, “apple” etc. Web nowadays is broken, huge unstructured data is available on web, mostly unusable. So there is a need to embed semantics to the web making it usable.

Technologies such as RDF/XML, OWL and Microformats are used for describing information and resources on the web. Just like metadata, RDF and OWL adds machine understandable data to the objects hidden from end user. Understanding the need of semantic web has led to the development of many projects in various domains including Semantic Web Services [**12**] [**13**] [**14**], Solving Cold Start Problems [**15**], Solving Scientific issues [**16**], Business Process Management [**17**], Biomedical [**18**], Semantics in Cloud [**19**] etc. Also various search engines have evolved providing semantic image [**20**] and web search results [**21**] such as

- **Hakia** [**22**]- A general purpose semantic search engine that search structured corpora (text) like Wikipedia.
    - In Hakia search results are organized in tabs say:
        - Web results, credible sites (sites that have been vetted by librarians)
        - Images

        - News etc.
    - For popular queries and queries where there is ambiguity, Hakia produces resumes. Every resume has an index of links to the information presented on the page for quick reference. The elements of these resumes vary according to the nature of the query (e.g. biography, bibliography, timeline etc. for persons, government, economy, culture etc. for countries). Resumes are excellent for researching a topic.
    - It is in the beta version.

- **SenseBot** [**23**] - It is a web search engine that summarizes search results into one concise digest on the topic of your query.
    - It attempts to understand what the result pages are about.
        - For this purpose it uses text mining to analyse web pages and identify their key semantic concepts.
    - Thus we do not have to go through a large number of web pages and comb through results with incomprehensible expert definitions.
    - Not all of the summaries are informative or even intelligible, but that is likely to improve; Like Hakia, SenseBot is in beta.

- **Powerset** – The Microsoft-acquired search engine Powerset (now merged to bing.com) focuses on doing only one thing and doing it really well. All search results on Powerset come from Wikipedia, making it the ultimate way to search Wikipedia, using semantics.
    - It offers a comprehensive view of information.
    - We can test it on Wikipedia, structuring the information and presenting it in a way useful for research purposes and is a great improvement on Wikipedia's own search engine.
    - Simple keywords, phrases, or simple questions in the search box. On the search results page, Powerset often answers questions directly. It aggregates information from across multiple articles.
    - "Factz" is a box that often appears in the search results and is a set of suggestions for reference queries based on the information available.

        - Say, when I search for Obama, Powerset offers links to information on what Obama has said about Middle East, Pakistan, trade and more. Clicking one of these links brings up a box in the search results page with the actual words said by Obama and links to the articles in which the quotes appeared.

- **DeepDyve** [**24**] - It is a powerful, professional research tool available for free for the general public.
    - It is a research engine that lets you access expert content from the "Deep Web", the part of the Internet that is not indexed by traditional search engines (e.g. databases, journals etc.).
    - Researchers, students, technical professionals, business users, and other information consumers can search Wikipedia or deep web resources within following categories:
        - Life Sciences and Medical
        - Physical Sciences
        - Humanities and Social Sciences
        - Business and Finance
        - Patents
        - Legal
        - Clean Technology and Energy
        - IT and Engineering.
    - Research sites' search engines often rely on Boolean languages or hard-coded taxonomies, which constitutes a threshold and makes them hard to use (or even inaccessible) to anyone but insiders.
    - Query can consist of anything from a single word to 25 000 characters.
    - The search results are presented in a complex manner with many advanced options for refining, sorting or saving a search.
        - Despite the complexity, the search results are relatively easy to navigate.

- **Cognition** [**25**]– It has a search business based on a semantic map, built over the past 24 years, which the company claims is the most comprehensive and complete map of the English language available today. It is used in support of business analytics, machine translation, document search, context search, and much more.
    - Cognition's technology to search one of following bodies of information:
        - Public.Resource.org (currently 1,858 volumes consisting of 675,704 files of federal case law in XHTML format). The release comprises US Supreme Court Decisions and Court of Appeals decisions from 1950 on.
        - MEDLINE (Medical Literature Analysis and Retrieval System Online)
        - Abstracts: abstracts for life sciences and biomedical information from an international literature database. It covers the fields of medicine, nursing, pharmacy, dentistry, veterinary medicine, and health care, as well as fields with no direct medical connection, such as molecular evolution (currently 18,005,903 files).
        - The English version of Wikipedia
    - Cognition is especially useful for sorting out meaning in complex queries:
        - Phrases like "historical houses of worship and historical temples"
        - Meaning: "worker on strike" vs. "strike oil in mathura"
    - Cognition gives you valuable control solving such queries by assigning meaning and classes in a user friendly way.

- **Swoogle** [**26**]- It is strictly for the semantic web.
    - The engine indexes documents developed on the concepts and standards for semantics (such as the RDF Format).
    - The aim of the engine is to return meaningful sentences for the search query.

These search engines and other semantic search engines mostly experimental extract the semantic data from webpages and search is performed on the basis of semantic data and not on simple pattern matching. This sound really good however RDF and OWL are effectively invisible metadata on the Web_i. Instead of adding data usable by the end user, it imposes an additional burden of adding invisible machine understandable data (semantics) to web hidden from end user. For illustration if you go to an RDF file and you actually try to view it, the browser will either download it for you or it will give you a bunch of gibberish.

Websites are meant for users, data on web is for users/authenticated by the users. If any data is not visible to the end user, there's a high probability it could be ignored or exploited. Few years back metadata came to picture for adding semantic data to the webpage. It failed, and the reason for the failure was quite obvious – metadata embedded to the webpage was invisible data, hidden from the end users.

Thus common issues with metadata are:

- Less descriptive or incomplete metadata.
- Incorrectly / deceivingly described metadata to gain more traffic.
- It is not regularly updating.
- No longer caring about metadata.
- Creator or author is different from the creator of visible data resulting indiscrepancy.

Above said reasons for the failure of metadata being invisible web applicable to RDF/OWL are quite obvious as it does not affect the users and is not authenticated by them. There's a need to use some methods to embed semantic data to the web in such a way to overcome above written pitfalls and lead to the success of the semantic web i.e. method that embeds semantics to the web along with visible data. One such solution is Microformats.

Microformats are a set of simple, open data formats built upon existing and widely adopted standards. Instead of throwing away what works today, microformats intend to

solve simpler problems first by adapting to current behaviours and usage patterns (e.g. XHTML, blogging) [**4**]. They are items of semantic mark-up, using just standard "plain old semantic HTML" (i.e. "POSH [**27**]") with a set of common class-names and "rel" values. They are open and available, freely, for anyone to use. Thus instead of changing how world works, it adjusts itself to coupe up with them adding additional functionality.

For an example:

An event on December 1st, 2012 of Jaguar at Ludhiana, Punjab, India is written in HTML as:

December 1st to December 31th, 2012 Jaguar Show at Ludhiana, Punjab, India

Set your pulse racing when you experience the sheer power and performance that is Jaguar for just £275 per person.

- accelerate hard down the straight and hear the engine roar

- enjoy the ease of controls as you test the car to its limits

- see how capable the driving dynamics are on our ice road

- feel the G forces at work as you break into the tight bends

Above written event would be written in HTML 4 and above:

<abbr title="2012-12-1">*Dec 1st*</abbr> to <abbr title="2012-12-31">*Dec, 31th*</abbr>*Jaguar Show at Ludhiana, Punjab, India*

Set your pulse racing when you experience the sheer power and performance that is Jaguar for just £275 per person.

- accelerate hard down the straight and hear the engine roar

- enjoy the ease of controls as you test the car to its limits

- see how capable the driving dynamics are on our ice road

- feel the G forces at work as you break into the tight bends

Above written event would be written in Microformats as:

```
<div id="hcalendar-Jaguar-Shows" class="vevent">
<abbr title="2012-12-30" class="dtstart">Dec 30th</abbr> to <abbr title="2012-12-31" class="dtend">Dec, 31th</abbr><span class="summary">Jaguar Show</span> at<span class="location">Ludhiana, Punjab, India</span>
<div class="description">
<p>
        Set your pulse racing when you experience the sheer power and performance that is Jaguar for just £275 per person.
<br />
      - accelerate hard down the straight and hear the engine roar
<br />
      - enjoy the ease of controls as you test the car to its limits
<br />
- see how capable the driving dynamics are on our ice road
<br />
      - feel the G forces at work as you brake into the tight bends
         </p>
</div>
</div>
```

As we could notice the difference, HTML code could simply be converted to the microformats code just by adding relevant predefined classes and rel tags.

In the above said case:

```
<div id="hcalendar-Jaguar-Shows" class="vevent">
```

It illustrates the beginning of calendar event with a descriptive name of “hcalendar-Jaguar-Shows”.

Here id=”hcalendar-Jaguar-Shows” indicates the division name and class=”vevent” [**28**] shows that it is a calendar event.

```
<abbr title="2012-12-30" class="dtstart">Dec 30th</abbr>
```

It shows the start of the event and abbreviation of Dec 30th is the date 2012-12-30 (YYYY-MM-DD) in numeric international date exchange format [**29**].

Here <abbr title=2012-12-30>Dec 30th</abbr> [**30**] shows the abbreviation for Dec 30 for search engines index.

class=”vevent” shows that it is a calendar event [**28**].

Ludhiana, Punjab, India

Above line shows that the tag contains location address. It could further be divided for more accurate location tagging [**31**] [**32**] [**33**].

Class name “description” illustrates that the division contains the description of the event and not anything else.

Finally we close all the open tags. We have seen how easily and efficiently we have been able to embed semantics to the existing html code with no or very little extra effort[2].We are interested to exploit microformats this property to embed semantics to the web pages and search engines to display relevant results on the basis of the tags describing data.

In this chapter we saw the working of current search engines, common issues faced by them, some ambiguous terms illustrating the limitation of search engines. Then we concluded the need for new technologies followed by discussing popular new semantic technologies in use; followed by their usage and applications, limitations. Then counter comparing them with each other. Then we discuss popular new semantic search

[2] Various tools are also available for producing microformats [**6**].

engines; discussed their working, strengths, weaknesses etc. Finally we illustrated the same with the help of an example. Now it's time to jot down our observations and the methodology to overcome them.

Chapter 3
Problem Formulation

While doing the research work following points are mainly observed on which work is required to be done.

3.1 Research Findings

I. Current web is more like a parrot; it does not understand the information but just mimics the information.

II. Search engines are working on the basis of pattern matching. So say it finds no difference between "Apple as a fruit" or say "Apple as a company".

III. Size of web has grown and is growing exponentially thus such ambiguity has also increased proportionally thus making it difficult for the search engines to find relevant results.

IV. Finding right information on web is more of being lucky then logical.

V. More popular things are given preference thus it gets hectic and time consuming to find less popular things with same name.

VI. Search engines do not understand your domain of interest and returns information from almost all fields or say more popular field(s).

VII. Results shown are independent and are not inter-related. Thus say if you search "I want to go for movie followed by dinner" as a search query, it may returns you results with cinema nearby and a restaurant say thousands of kilometres away from the cinema it is proposing.

3.2 Problem Statement

In this thesis work is done to bridge some of the above said observations which includes web understanding the meaning behind the information by adding relevant tags to the information, search engines reading those tags and providing few but relevant results. Finally efficiency of the proposed system is evaluated by comparing its results with other existing popular search engines such as Google and/or Yahoo/Bing etc.

3.3 Objectives

The main objectives of the proposed work are:

I. To study and analyse the working of current search engines to make them more efficient.
II. To propose a method to make the search engines more efficient.
III. To develop a search engine to extract semantically relevant results.
IV. To prove the accuracy of the proposed search engine.

3.4 Methodology

To achieve the above said goal, step by step methodology has been followed. The details of the same are given below:

I. Current search engines are working on pattern matching. Microformats tags are proposed to make webpages semantically understandable.
II. Search engine should perform search on the basis of microformats tags instead of simple pattern matching.
III. A) A website is specially designed embedded with microformats.
B) Search engine is developed to perform semantic search on this website.
IV. Accuracy of the search engine is evaluated by designing test cases which are made to run on our microformats based search engine and regular search engines such Google and/or Yahoo/Bing etc.

Chapter 4

Exploration of website Design Tools

A website is specially designed to illustrate the efficiency of Microformats based search engine. Website is then uploaded to the live webserver (www.microformats.co.nr) for search engine index. Then structured data is extracted from the webpages by the search engine to perform semantic search.

HTML, CSS and JavaScript are being used as the fundamental technologies for building the Web pages. Where HTML [**34**] (html and xhtml) for the structure of the document — by denoting certain text as headings, paragraphs, lists etc. and it supplement that text with interactive forms, embedded images, and other objects. Cascade Style Sheets (CSS) [**34**]are used for providing style and layout. JavaScript [**35**] is being used for adding dynamic content to the webpages. And to make our job easier, Adobe Dreamweaver is being used for assistance doing all the above work quickly and efficiently. Finally Adobe Photoshop is being used as image processing software.

Technology used:

- HTML/XHTML [**34**]
- CSS [**34**]
- JavaScript [**35**]

Tools used:

- Adobe Photoshop CS5 [**36**]
- Adobe Dreamweaver CS5 [**37**]
- Filezilla FTP client 3.5.3 [**38**]
- Apache Webserver [**39**]

4.1 Technology used

4.1.1 HTML/XHTML

HTML is an acronym for "Hyper Text Markup Language". It uses markup tags to describe webpages, thus is called markup language and not a programming language. HTML tags are keywords surrounded by angle brackets e.g. <html>. Web browsers such as IE (Internet Explorer), Firefox, Safari, Chrome etc. read HTML documents and display them as web pages. It does not display html tags but uses those tags to interpret the content of the webpages.

Example:

```
<html>
<head>
<title>HTML demonstration page</title>
</head>
<body>
<b>
Hi !! <br />
You could notice how clearly browser ignores HTML markup tags and displays
formatted content on the basis of tags.
</b>
</body>
</html>
```

Above shows markup is a simple webpage with title "Microformats based search engine" displaying output in Figure 4.1. Here we could notice, browser has ignored html markup tags but displayed formatted output on the basis of tags. The <html> tag tells the browser that this is an HTML document. This is called root element. It is container for all other HTML elements (except for the<!DOCTYPE> tag).

The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in [**40**].

The <head> element is a container for all the head elements. It must include a title for the document and can include scripts, styles, Meta information and more [**41**].

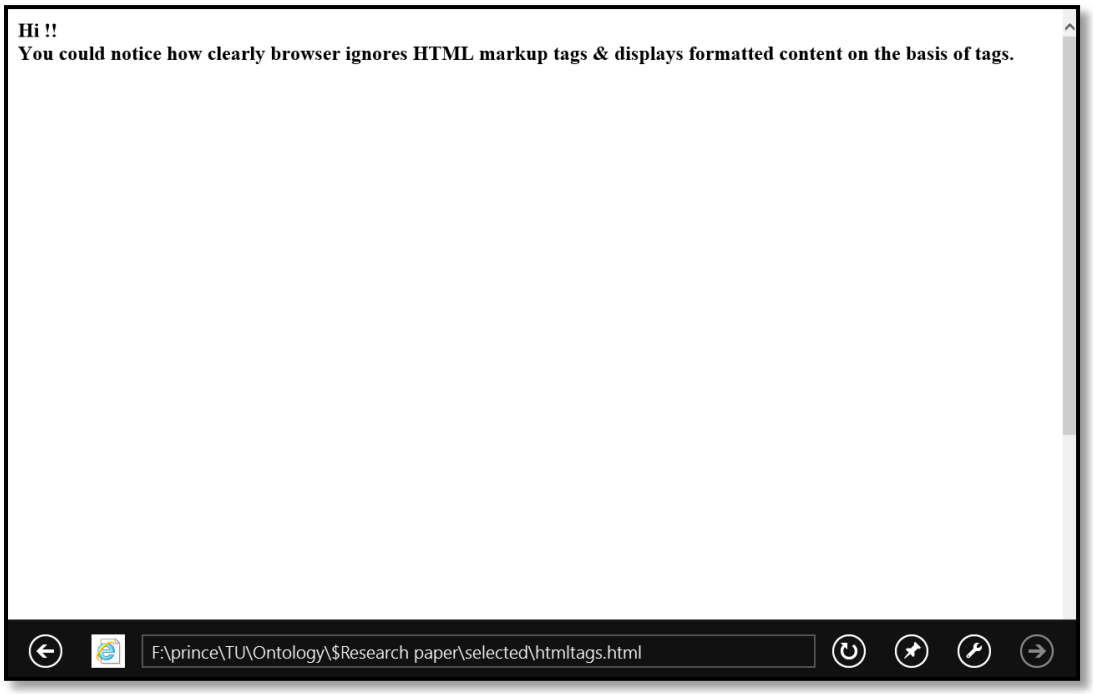

Figure 4.1: Webpage showing ouput of above written HTML markup

<head> element can include following elements:

- <title> (this element is required in the head section)
- <style>
- <base>
- <link>
- <meta>
- <script>
- <noscript>

The <title> tag defines the title of the document. It is required in all HTML/XHTML documents [**42**].

The <title> element

- defines a title in the browser toolbar
- provides a title for the page when it is added to favorites
- displays a title for the page in search-engine results

The <body> tag defines the document's body. The <body> element contains all the contents of an HTML document, such as text, hyperlinks, images, tables, lists, etc. [**43**]

The <b> tag specifies **bold text** [**44**].

4.1.2 Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents [**45**]. Styles define "how to display" HTML elements. CSS solved a big problem. HTML was never intended to contain tags for formatting a document.

HTML was intended to define the content of a document, like:

<h1>This is a heading</h1>
This is a paragraph.

However when tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large web sites, where fonts and color information were added to every single page, became a long and expensive process. To solve this problem, the World Wide Web Consortium (W3C) created CSS.

In HTML 4.0, all formatting has been removed from the HTML document, and stored in a separate CSS file which saves lot of work. All browsers support CSS today [**46**].

4.1.3 JavaScript (JS)

JavaScript is ideally used as client scripting language designed to add interactivity to HTML pages. JavaScript is usually embedded into HTML pages. It is an interpreted language (means that scripts execute without preliminary compilation) [**47**].

Features of JavaScript:

- It gives HTML designers a programming tool. It has very simple syntax. Almost anyone can put small snippets of code into their HTML pages.
- It can respond to events such as user click, page load etc.

- JavaScript can read and write HTML elements i.e. it can read and change the content HTML element.
- It can be used to validate data before it is submitted to a server. This saves server from extra processing.
- It can detect the visitor's browser and thus could perform browser centric operations.
- It can create cookies.

The HTML <script> tag is used to insert a JavaScript into an HTML page.

Example:

```
<html>
<body>
<h1>Embedding JavaScript to HTML - Thapar University, Patiala</h1>
<script type="text/javascript">
document.write("<p>" + Date() + "</p>");
</script>

</body>
</html>
```

Output of the above written code is being displayed in Figure 4.3. Here JavaScript shows client's system time and date with its time zone. As we have seen that JavaScript runs on client and could gain access to the client's system. Thus in some cases JavaScript is being blocked in some browsers as shown in Figure 4.2 where the above written HTML code embedded with JavaScript is being executed on Internet Explorer 10 (IE10) metro version.

It has blocked the JavaScript on the webpage and seeks for permission. Clicking on "Allow blocked content" would allow JavaScript to be executed and displays the output on the webpage as shown in Figure 4.3.

Most browsers display the prompt whenever the JavaScript content is blocked and allows you to execute blocked content (scripts). However otherwise if required JavaScript can unblocked permanently* on the web browser by following steps [**48**].

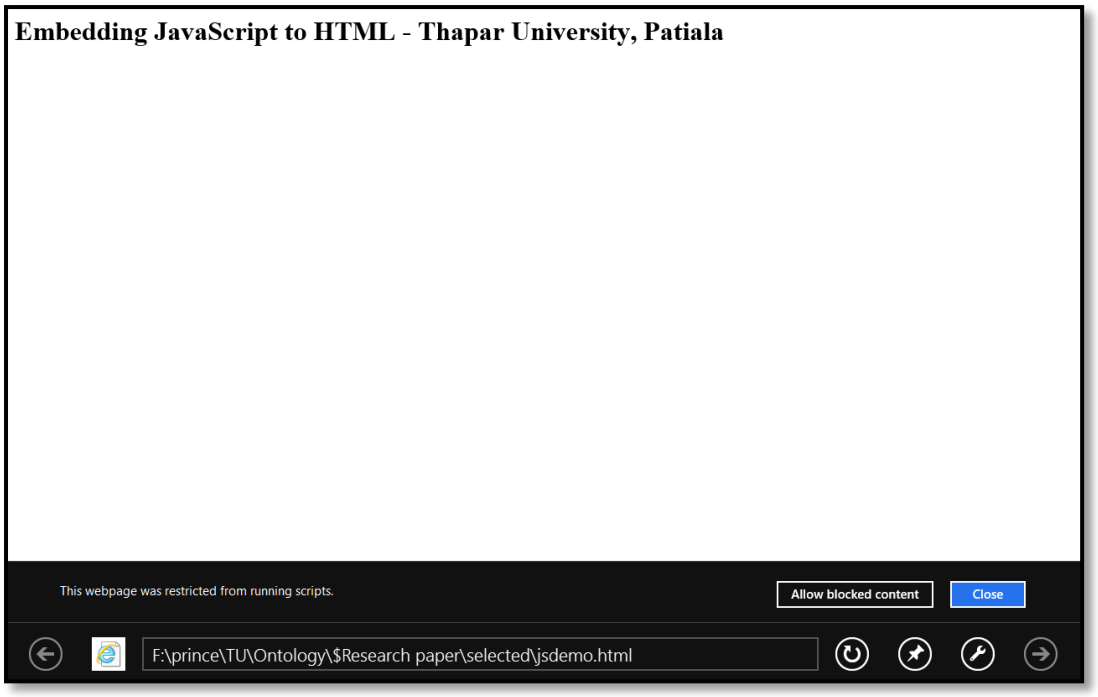

Figure 4.2: JavaScript restricted on webpage

Google Chrome (15.0)

1. Click the spanner icon on the browser toolbar.
2. Select **Options**.
3. Click the **Under the Hood** tab.
4. Click **Content Settings** in the 'Privacy section.'
5. Select **Allow all sites to run JavaScript** in the 'JavaScript' section.

Mozilla Firefox (8.0)

1. Select **Tools** from the top menu.
2. Choose **Options**.
3. Choose **Content** from the top navigation.
4. Select the checkbox next to **Enable JavaScript** and click **OK**.

Internet Explorer (9.0)

1. Select **Tools** from the top menu.
2. Choose Internet Options.

3. Click on the **Security** tab.
4. Click on Custom Level.
5. Scroll down until you see the section labeled 'Scripting.'
6. Under 'Active Scripting,' select **Enable** and click **OK**.

Apple Safari (5.0)

1. Open the **Safari** menu on your browser's toolbar.
2. Choose Preferences.
3. Choose **Security**.
4. Select the checkbox next to **Enable JavaScript**.

For further details to enable JavaScript on browser is available here [**49**].

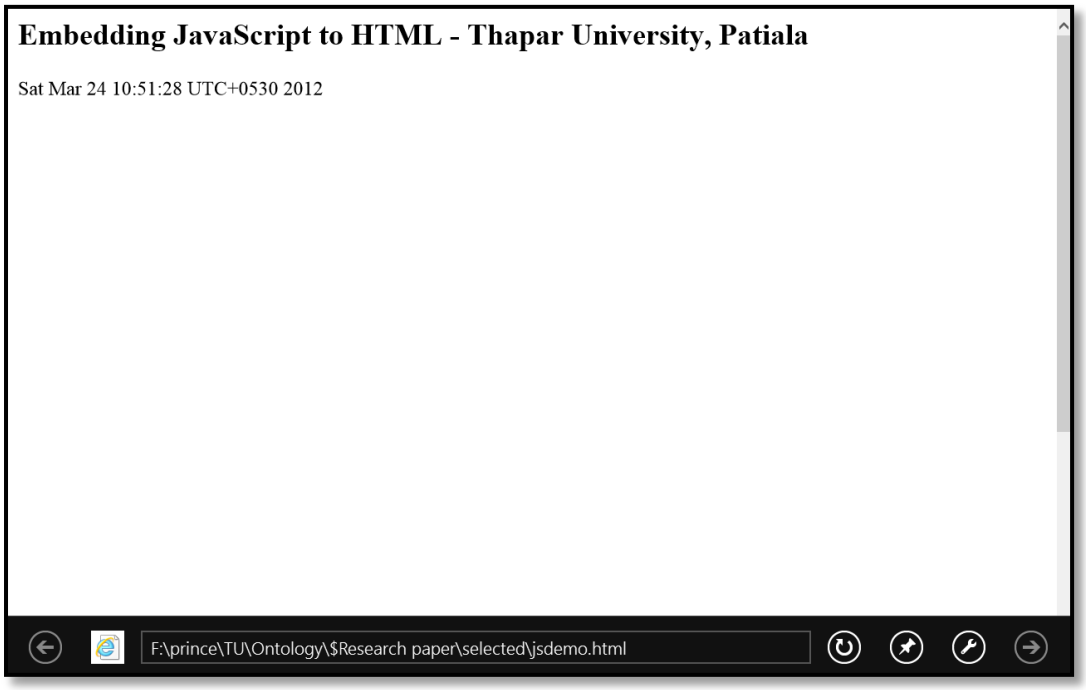

Figure 4.3: JavaScript Showing current date and time

4.2 Tools Used

4.2.1 Image basics (Adobe Photoshop CS5)

Every bitmap is made up of very small dots as shown in Figure 4.4 called pixels which constitute the resolution of the picture.

There are four ways to measure resolution:

- LPI: Lines per inch (Figure 4.5)
- SPI: Spots per inch (Figure 4.5)
- DPI: Dots per inch
- PPI: Pixels per inch

Modifying/creating these pixels actually modifies/creates the image called image processing.

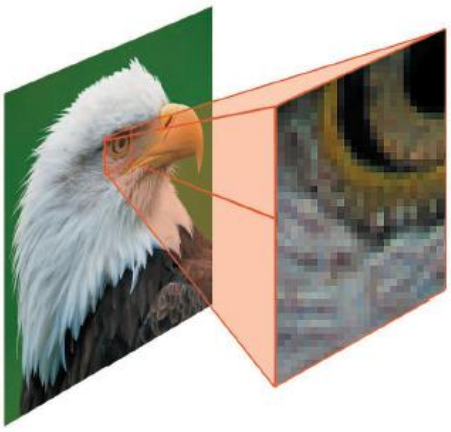

Figure 4.4: How dots (pixel) forms the image

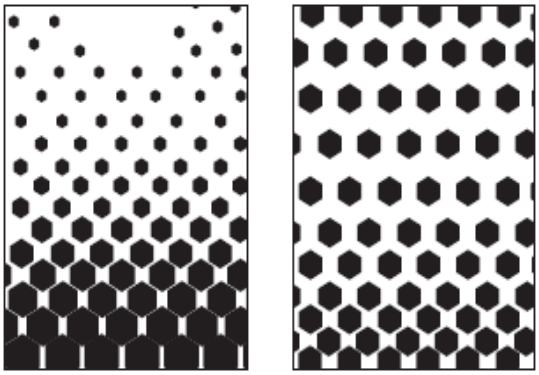

Figure 4.5: Different dot patters in graphics

4.2.1.1 Image Processing

Image processing is a method to convert an image into digital form and to perform some operations on it, in order to enhance image or to extract some useful information from it. Here input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional applying operations on them.

Image processing basically includes the following three steps:

- Importing the image say by optical scanner or by digital photography.

- Analyzing and manipulating the image which includes data compression and image enhancement etc.
- In the last stage result can be altered image or report that is based on image analysis.

Purpose of Image processing:

- Visualization - Observe the objects that are not visible.
- Image sharpening and restoration - To create a better image.
- Image retrieval - Seek for the image of interest.
- Measurement of pattern – Measures various objects in an image.
- Image Recognition – Distinguish the objects in an image.

Adobe Photoshop is a dominant image-editing program and electronic painting software program in the world. Whether we are working on a webpage, PowerPoint presentation or a document to be printed, Photoshop can be used to enhance our images. And thus the latest version [3] Adobe Photoshop CS5 [**36**] has been used to create/edit/enhance the images used in our website.

4.2.1.2 Adobe Dreamweaver Creative Suite 5 (CS5)

Adobe® Dreamweaver® CS5 is the industry-leading web authoring and editing software that provides both visual and code-level capabilities for creating standards-based websites and designs for the desktop, smartphones, tablets, and other devices [**37**].

Adobe Dreamweaver CS5 has been used to assist through the development process providing both visual and code level editing capabilities supporting CSS3. Thus saving development time by reducing overhead of remembering all the tags which otherwise are not required working in visual mode.

---

[3] As on January, 2012

4.2.2 Filezilla FTP client

FTP - File Transfer Protocol is the name of the application and the protocol used for moving files between two hosts on a TCP/IP network.

Filezilla FTP client Client 3.5.3 [**38**]is being used to transfer files/website documents to/from the webserver. FileZilla is an open source software distributed free of charge under the terms of the GNU General Public License [**50**].

4.2.3 Apache Webserver

A Web Server is a Computer or Combination of computers, which is connected through internet or intranet to serve the clients quests, coming from their web browser. It is like a large repository of web pages which respond to client requests.

We can also define the web server as the package of large number of programs installed on a computer connected to Internet or intranet for downloading the requested files through protocol such as File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP) etc. for their own specific use.

A web server works on a client server model. A computer connected to the Internet or intranet must have a server program. For an example if we are talking about Java language then a web server is a server that is used to support the web component like the Servlet and JSP.

We have used Apache Webserver 2.4supporting HTML4 for publishing website and associating a domain name (microformats.co.nr) to it.

Chapter 5

Website and Search Engine Structure

As previously discussed a website is specially designed to illustrate the power of microformats in search engine. In this website different types of microformats are being embedded such as microformats showing personal information, Resume, Events, Views, House/organization address, Products, Bulletins, Advertisement, Media etc. and to query above said information, a search engine is design. Google CSE XML API's are being used to power the search engine and to illustrate the real power of Microformats search, same keywords are being used in the website in different context on the different web pages other than intended giving search engine a real environment where an ordinary search engine working on the basis of pattern matching would fail however search engine working on the basis of microformats would succeed.

Google search engine is then used to index the website for structured and normal search; it powers the search engine with real required algorithms. Finally efficiency of the search engine is evaluated via specially designed test cases. Test cases are made to run both the microformats based search engine and normal search engines such as Google and/or Yahoo/Bing etc.

5.1 Layered Architecture of Website

Specially designed website is implemented at domain www.microformats.co.nr. Layered diagram in Figure 5.1 shows the basic architecture of microformats based website.

Here, data is embedded to the website via XHTML. Then microformats tags are being used to display information of different domains called elementary microformats. An elementary microformat can be enclosed into other microformats forming compound microformats. Thus, a compound microformat is a combination of elementary microformats and/or standard HTML elements [**51**]. hCalendar, hCard and hReview are few popularly used compound microformats. Query Engine implemented via Google CSE XML APIs is being used to index these embedded microformats and to perform

semantic search on the basis of extracted structured data. Query is being performed and results are displayed on the web browser such as Firefox, Internet Explorer, Safari, Google Chrome, Opera etc.

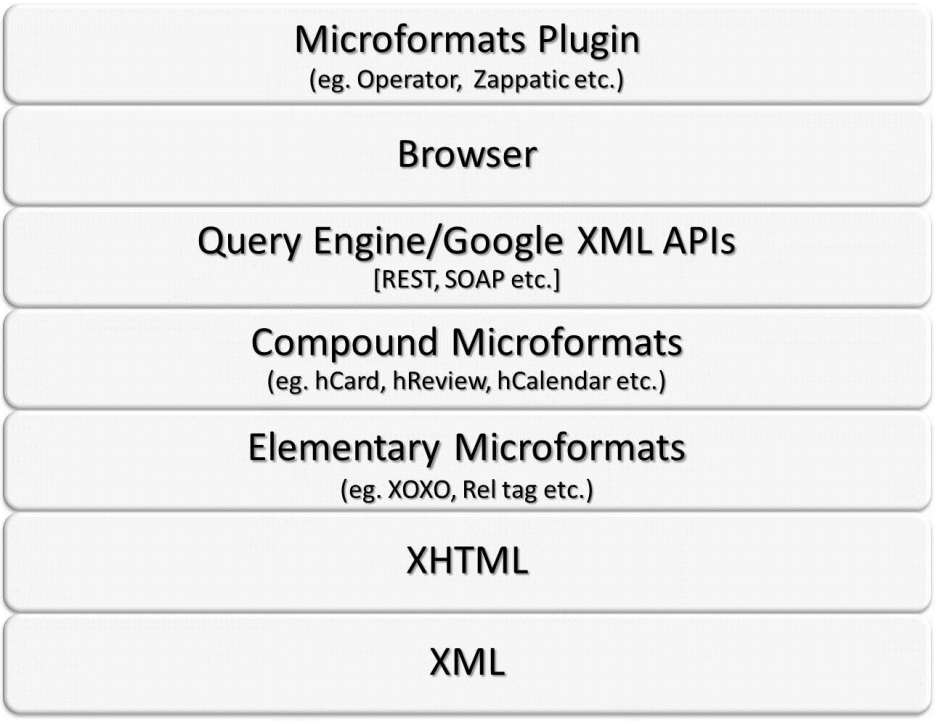

Figure 5.1: Layered architecture of Microformats based website

5.2 Microformats Browser Plugins

Various microformats plugins are available for different browsers providing extra microformats features on the basis of embedded microformat such geo microformat supports finding the directions via Google/Bing maps by the distance of fewer clicks. For an example hProduct supports finding similar products on amazon, ebay etc. Number of additional features provided by Microformats plugin varies from plugin to plugin limited by the web browser it supports. Various microformats plugins supported by different browsers are shown in Table 5.1.

Table 5.1: List of available Microformats Plugins

Plugin Name	**Supported Browser**

Operator [**52**]	Mozilla Firefox
Oomph [**53**]	Internet Explorer
Zappatic [**54**]	Apple - Safari
Microformats for Google Chrome™ [**55**]	Google Chrome

Microformats based query/search engine extracts the structured data after validation from the Microformats embedded in the webpages then structured query is executed on the extracted structured data and returned to the browser running search engine application.

5.3 Search Engine Structure and Process

Structure of Microformats based search engine in shown in Figure 5.2.

Here,

1. Data is embedded in XHTML using microformats tags.
2. These web pages are being parsed and indexed after validation.
    a. In the parsing phase structured data and search keywords are being collected separately.
3. Index repository saves the search keywords and structured data separately.
4. This saved information is being queried by the search engine to find relevant results.
    a. Search engine first perform a simple pattern matching search querying relevant keywords and there synonyms.
    b. Then results are filtered according to the relevant microformats tags.
5. Finally search results are returned to the browser/search engine application from where query was initiated.
6. Microformats plugins as shown in Table are used to explore additional user centric capabilities of microformats.

Process of extracting a microformat both by query/search engine and a microformats plugin is being shown in detail in Figure 5.3 in the form of a flow chart.

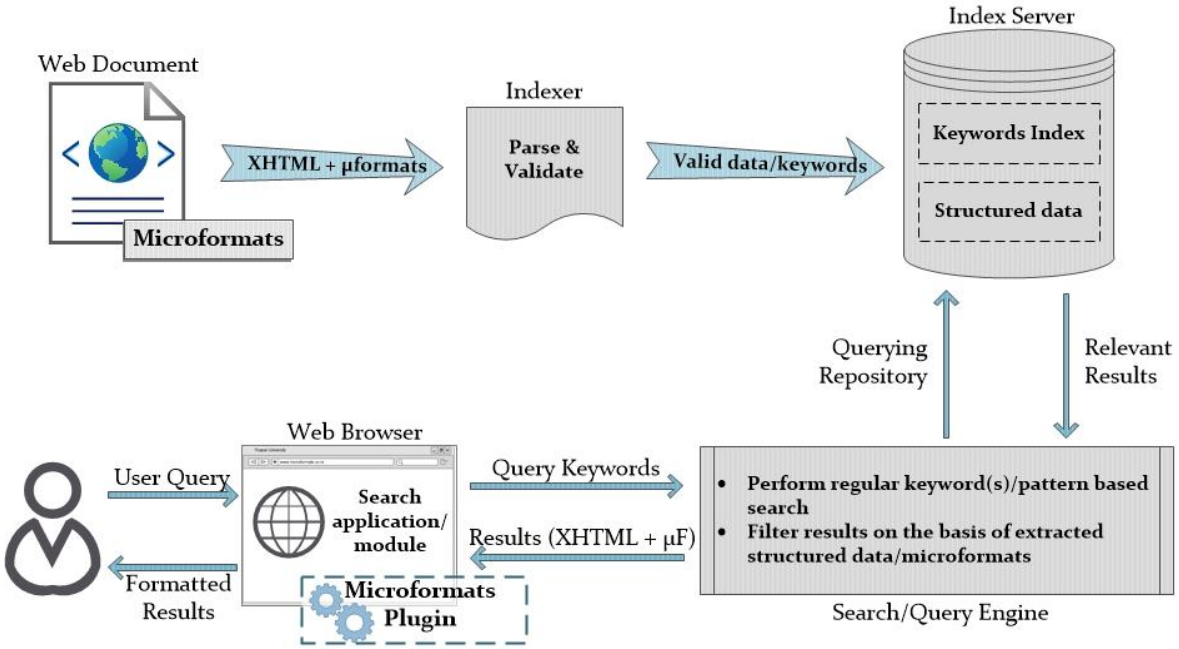

Figure 5.2: Microformats based search engine structure and process

Extracting a microformat from a webpages has following steps:

1. Initially Root Element is searched. E.g. <class> <a> etc.
2. If a Root Element is found.
    a. It is checked for the valid Microformat class attribute say hCard, hCalendar, rel etc.
    b. If valid microformat class attribute is not found then it's not a microformat and is just a simple XHTML element so it would be read as normal element.
3. If Valid Microformat class attribute is found, it is checked for the enclosed text.
    a. If inner text is found, element is parsed forming elementary microformat.
4. If there is no inner text then it must be a compound microformat, a check is being performed to look for inner element/node forming elementary microformat.
    a. If inner element/node is found, control is sent back to check if it is a valid microformat class attribute forming an elementary microformat.

    b. Same process is repeated until all valid microformats are parsed.
5. When no more elementary microformats are being found, it's time to concatenate them; if compound microformat.
6. Check is made to check if a microformat contains elementary microformats.
7. If there are elementary microformats, they are concatenated to form compound microformat.
8. Finally microformat is successfully extracted from the webpage and optionally a subroutine is called to extract structured data from the extracted microformat; in case the above said process is being followed by Index server.

5.4 Popular Microformats

Various Microformats embedded in our website are:

- Calendar Event (vevent)
- People and organizations (vCard)
- Reviews (hReview)
- Products (hProduct)
- Vote Links (vote-for, vote-abstain, vote-against)
- Audio (Microdata)
- Movie (Microdata)
- News (hNews)
- Open XHTML Outlines (xoxo)
- Resume (hResume)
- License (rel-license)

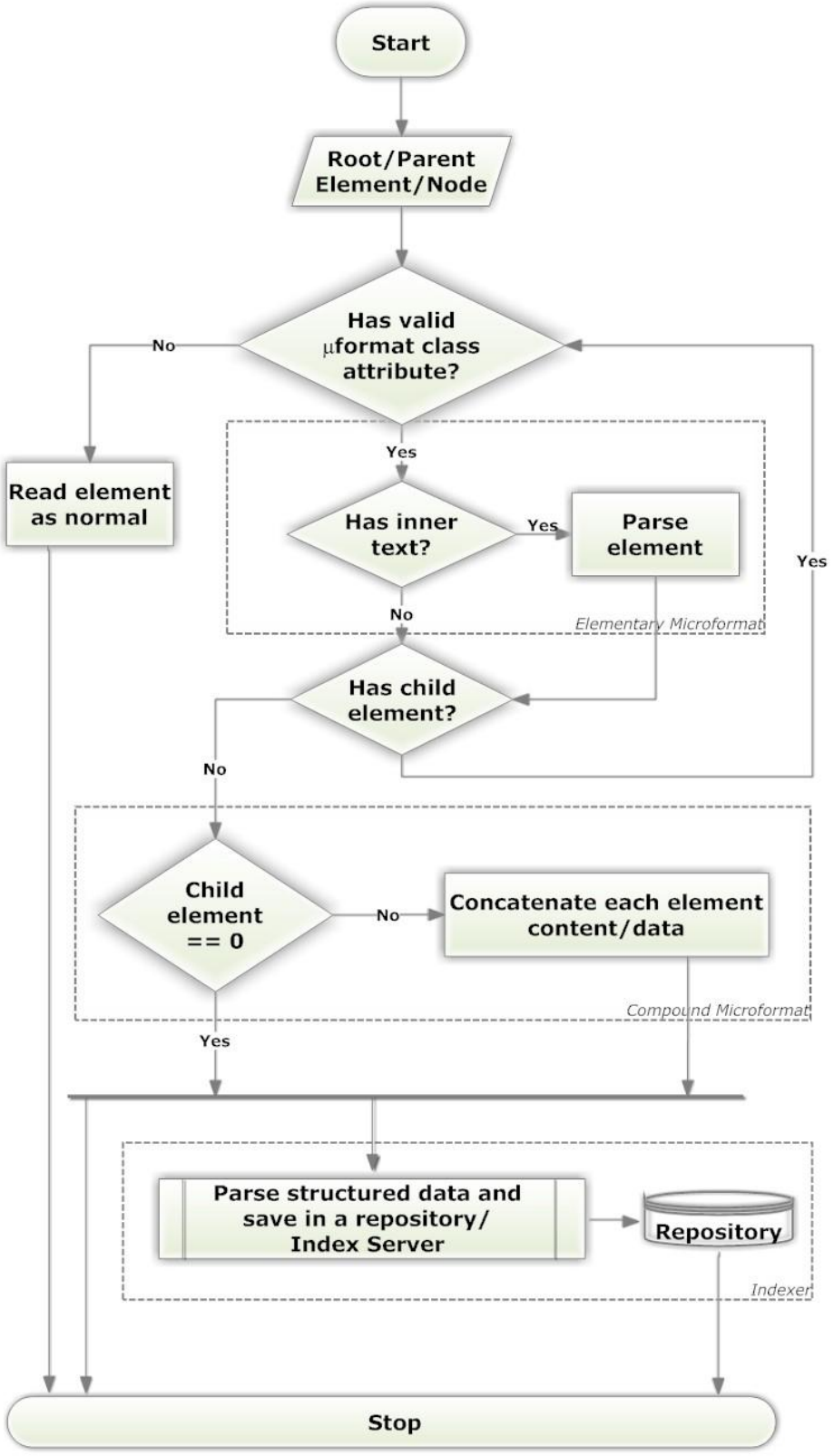

Figure 5.3: Process of extracting a Microformat from a webpage

5.4.1 Calendar Event (vevent)

hCalendar is a simple, open, distributed calendaring and events format, using a 1:1 representation of standard iCalendar (RFC2445) VEVENT properties and values in semantic HTML or XHTML [**28**].

Calendar event embedded in our website:

```
<div class="vevent">
<a href="http://llrcp.ac.in/microformats/events/spinaltap" class="url summary">Spinal Tap</a>
<img src="spinal_tap.jpg" class="photo" />
<span class="description">After their highly-publicized search for a new drummer, Spinal Tap kicks off their latest comeback tourwith a San Francisco show.</span>
  When:
<span class="dtstart">
    Oct 15, 7:00PM<span class="value-title" title="2015-10-15T19:00-08:00"></span>
</span>-
<span class="dtend">
    9:00PM<span class="value-title" title="2015-10-15T21:00-08:00"></span>
</span>
  Where:
<div class="location vcard">
<span class="fn org">Warfield Theatre</span>,
<span class="adr">
<span class="street-address">982 Market St</span>,
<span class="locality">San Francisco</span>,
<span class="region">CA</span>
</span>
<span class="geo">
<span class="latitude">
<span class="value-title" title="37.774929" ></span>
</span>
<span class="longitude">
<span class="value-title" title="-122.419416"></span>
</span>
</span>
</div>
  Category: <span class="category">Concert</span>
</div>
```

Structured data extracted from the above snippet:

hcalendar

date start = 2015-10-15T19:00-08:00
summary = Spinal Tap
location
hcard
fn = Warfield Theatre
org
organization-name = Warfield Theatre
adr
street-address = 982 Market St
locality = San Francisco
region = CA
url = http://llrcp.ac.in/microformats/events/spinaltap
date end = 2015-10-15T21:00-08:00
category = Concert
description = After their highly-publicized search for a new drummer, Spinal Tap kicks off their latest comeback tour with a San Francisco show.
photo = http://microformats.co.nr/spinal_tap.jpg
geo
latitude = 37.774929
longitude = -122.419416

Above snippet shows that it's a calendar event starting on 15th October, 2015 at 7pm to 8am with subject/summary "Spinal Tap" location is at Warfield Theatre which is embedded via vCard microformat. Finally complete description an event is being shown in the description field.

5.4.2 People and organizations (vCard)

hCard is a simple, open, distributed format for representing people, companies, organizations, and places, using a 1:1 representation of vCard (RFC2426) properties and values in semantic HTML or XHTML [**56**].

vCard embedded in our website:

```
<div id="hcard-Jaguar-Saini" class="vcard">
<img style="float:left; margin-right:4px"
src="http://a0.twimg.com/profile_images/490760078/rs.jpg" alt="photo of "
class="photo"/>
<span class="fn">Jaguar Saini</span>
<a class="email" href="mailto:test@testing.com">test@testing.com</a>
<div class="adr">
<div class="street-address">Kipps Market, Sarabha Nagar</div>
<span class="locality">Ludhiana</span>,
<span class="region">Punjab</span>,
```

141001
India

1234567890
Software Engineer at
Microsoft.
AIM
YIM
Jabber
<a
href="http://kitchen.technorati.com/contacts/tag/jaguar">jaguar<a
href="http://kitchen.technorati.com/contacts/tag/saini">saini This hCard created with the hCard creator.

Structured data extracted from above snippet:

hcard
fn = Jaguar Saini
n
family-name = Saini
given-name = Jaguar
org
organization-name = Microsoft
adr
street-address = Kipps Market, Sarabha Nagar
locality = Ludhiana
region = Punjab
postal-code = 141001
country-name = India
tel
value = 1234567890
email
value = test@testing.com
title = Software Engineer
url = aim:goim?screenname=jaguaraim
url = ymsgr:sendIM?jaguaryim
url = xmpp:jaguarjabber
photo = http://a0.twimg.com/profile_images/490760078/rs.jpg

Above snippet is describing a person “Jaguar Saini” working as a Software Engineer at Microsoft Corporation living in Ludhiana.

5.4.3 Reviews (hReview)

hReview is a simple, open, distributed format, suitable for embedding reviews (of products, services, businesses, events, etc.) in HTML, XHTML, Atom, RSS, and arbitrary XML [**57**].

hReview embedded in our website:

```
<div id="hreview-Shocking-departure-for-Jaguar" class="hreview">
<h2 class="summary">Shocking departure for Jaguar XJ Car</h2>
<abbr title="2008-06-30T23:32+05:30" class="dtreviewed">Jun 30, 2008</abbr>
 by
<span class="reviewer vcard"><span class="fn">Jonny Depp </span></span>
<span style="display: none;" class="type">product</span>
<img class="photo"
src="http://cache.daylife.com/imageserve/02Zp4oyatEek2/50x50.jpg" alt="photo of
'Jaguar'">
<div class="item">
<a href="http://www.carbuyer.co.uk/reviews/jaguar/xj/review?gclid=CP6y7_uq-
K0CFYkc6wod1mnMsg" class="fn url">Jaguar</a>
</div>
<abbr class="rating" title="4">★★★★☆</abbr>
<span style="display: none;" class="version">0.3</span>
<p>
Despite the shocking new look, the XJ shares one important link with the past… It's
built on the same lightweight aluminum chassis and body technology of its predecessor.
Long and short wheelbase versions are available, and the flagship 5.0-litre petrol engine
is available in two power levels. The 3.0-litre diesel, though, is our favorite. It returns
more than 40mpg - hardly fat cat consumption. Four specifications are available, aptly
called Luxury, Premium Luxury, Portfolio and Super sport.
</p>
</div>
```

Structured data extracted from above snippet:

```
hreview
version = 0.3
summary = Shocking departure for Jaguar XJ Car
type = product
item fn = Jaguar
url = http://www.carbuyer.co.uk/reviews/jaguar/xj/review?gclid=CP6y7_uq-
K0CFYkc6wod1mnMsg
reviewer hcard
fn = Jonny Depp
n
family-name = Depp
given-name = Jonny
dtreviewed = 2008-06-30T23:32+05:30
```

rating
value (normalized to 5.0 scale) = 4.0
value = 4

Above snippet shows a review of product “Jaguar” done on 30th June, 2008 by a person “Jonny Deep” whose information is being shown by vcard microformat giving rating of 4 on 5 point scale.

5.4.4 Products (hProduct)

hProduct is a microformat suitable for publishing and embedding product data. hProduct is one of several open microformats standards suitable for embedding in HTML, XHTML, Atom, RSS, and arbitrary XML [**58**].

hProduct microformat embedded in our website:

```
<div class="hProduct">
<h1 class="fn">Jaguar XFS IFB Washing machine</h1>
<p class="description">
      It's a joint venture of IFB and Jaguar International Inc.! It's is world's fastest clothes washing machine.
</p>
<p>
    Brand: <span class="brand">Jaguar</span><br />
    Price: Rs.<span class="price">29700</span><br />
<span class="reviewaggregate hreview-aggregate">
<span class="item">
<span class="fn">Rating:</span>
</span>
<span class="rating">3.5</span><br />
Reviews:<span class="count">87</span></span><br />
 Availability: <span class="availability in-stock">In stock</span><br />
Shipping time:
<span class="value-title" title="In stock">Can ship within 24 hours.</span><br />
<img src="images/washing.jpg" class="photo" width="200" height="137" alt="Jaguar IFB"/>
<h2>Specifications</h2>
<ul class="identifier">
<li class="type">
        Model: <span class="model">XFS</span>
</li>
<li>
                Capacity: <span class="value">8.5</span> liters
</li>
<li>
                Duration: <span class="value">1.5</span> hours
```


Click here for more details.

Structured data extracted from above snippet:

hproduct
availability = In stock
brand name = Jaguar
description = It's a joint venture of IFB and Jaguar International Inc.! It's is world's fastest clothes washing machine.
fn = Jaguar XFS IFB Washing machine
photo = http://llrcp.ac.in/microformats/images/washing.jpg
url = http://llrcp.ac.in/microformats/example.com/XFS
identifier
type = Model: XFS
value = 8.51.5
price = 29700
currency = INR
review hreview-aggregate
item fn = Rating:
rating
average (normalized to 5.0 scale) = 3.5
average = 3.5
count = 87

Above snippet is about a product “Jaguar XFS IFB Washing machine” with a rating of 3.5/5 available at Rs. 29700/-.

5.4.5 Vote Links (vote-for, vote-abstain, vote-against)

VoteLinks is an elemental microformat. We propose a set of three new values for the rev attribute of the <a> (hyperlink) tag in HTML.

The new values are "**vote-for**", "**vote-abstain**" or "**vote-against**", which are mutually exclusive, and represent agreement, abstention or indifference, and disagreement respectively.

A link without an explicit vote 'rev' value is deemed to have value "vote-for" or "vote-abstain", depending on the application.

Additional human-readable commentary can be added using the existing 'title' attribute, which most browsers show as a rollover [**59**].

Vote Links embedded in our website:

```
<dl class="profile">
<dt id="rev"></dt>
<dd><p>
  --><a rel="help" href="http://example.com"><strong>Additional
Help</strong></a><--<br />
<br />
<strong>Do you think jaguar XF is the best car in the world.</strong></p>
<dl>
<dt id="vote-for"><strong>Agree</strong></dt>
<dd>Yes, I support it!!</dd>
<dt id="vote-abstain"><strong>Can't Say</strong></dt>
<dd>Whatever, I don't care.</dd>
<dt id="vote-against"><strong>Disagree</strong></dt>
<dd>Highly Disagree!! <a href="http|//www.chevrolet.com/camaro-performance-
cars/">Chevrolet Camaro</a> is right for me.</dd>
</dl>
</dd>
</dl>
```

Above is a poll for Jaguar XF to be the best car in the world. We could notice here, how clearly it shows the favor, disagreement or abstain the vote.

5.4.6 Audio (Microdata)

An audio element represents a sound or audio stream. The audio element is a media element whose media data is ostensibly audio data. Common attributes to all media elements includes src, preload, autoplay, mediagroup, loop, muted, and controls attributes [**60**].

In Microformats, hAudio is used for embedding information about audio recordings in (X)HTML, Atom, RSS, and arbitrary XML [**61**].

Audio element embedded in your website (microdata):

```
<div itemscope itemtype="http://schema.org/MusicGroup">
<div class=”description”>
<div class="albumArt">
<a href="" title="You And I">
```

<img
src="http://cms.myspacecdn.com/cms/Music%20Vertical/Common/Images/default_large.jpg" alt="You And I">

Album: Jaguar You and I

Released: 23 May 2011
Songs:

<ul class="groupList trackList" >
<li data-songId="81953027" itemprop="tracks" itemscope
itemtype="http://schema.org/MusicRecording">
<h6><a href="/ladygaga/music/songs/you-and-i-81953027"
class="title">You And I</h6><span
itemprop="name">Born This Way<meta itemprop="duration"
content="PT5M7S" /><meta itemprop="url"
content="http://www.myspace.com/music/player?song=http%3a%2f%2fwww.myspace.com%2fladygaga%2fmusic%2fsongs%2fyou-and-i-81953027" /> 5:07

<li data-songId="81953027" itemprop="tracks" itemscope
itemtype="http://schema.org/MusicRecording">
<h6><a href="/ladygaga/music/songs/you-and-i-81953027"
class="title">The Edge Of Glory</h6><meta
itemprop="duration" content="PT5M20S" /><meta itemprop="url"
content="http://www.myspace.com/music/player?song=http%3a%2f%2fwww.myspace.com%2fladygaga%2fmusic%2fsongs%2fyou-and-i-81953027" /> 5:20

<li data-songId="81953027" itemprop="tracks" itemscope
itemtype="http://schema.org/MusicRecording">
<h6>
Born This Way</h6>
<meta itemprop="duration" content="PT4M20S" /><meta itemprop="url"
content="http://www.myspace.com%2fladygaga%2fmusic%2fsongs%2fborn-this-way-79776619" /> 4:20

Structured data extracted from above snippet:

Item 1
Type: http://schema.org/musicrecording
name = You And I
name = Born This Way
duration = PT5M7S
url =
http://www.myspace.com/music/player?song=http%3a%2f%2fwww.myspace.com%2fladygaga%2fmusic%2fsongs%2fyou-and-i-81953027
Item 2
Type: http://schema.org/musicrecording
name = The Edge Of Glory
duration = PT5M20S
url =
http://www.myspace.com/music/player?song=http%3a%2f%2fwww.myspace.com%2fladygaga%2fmusic%2fsongs%2fyou-and-i-81953027
Item 3
Type: http://schema.org/musicrecording
name = Born This Way
duration = PT4M20S
url = http://www.myspace.com%2fladygaga%2fmusic%2fsongs%2fborn-this-way-79776619

Above snippet is showing 3 tracks of Jaguar Wright:

- You and I 5:07
- The edge of glory 5:20
- Born this way 4:20

5.4.7 Movie (Microdata)

A movie element represents a sound or audio stream.

Audio element embedded in your website (microdata):
<div itemscope itemtype="http://schema.org/Movie">

<h1 class="header" itemprop="name">
Jaguar Inception
(2010)
</h1>
<div class="star-box giga-star" itemprop="aggregateRating" itemscope itemtype="http://schema.org/AggregateRating">

<img src="images/inception.jpg"
alt="Inception Poster" width="103"
height="146"
style="max-width:214px; max-height:317px;"
title="Inception Poster"
itemprop="image" />

<div class="rating rating-list" data-auth="BCYhVcyExvnAq9x_q2tpMM4ifc9bLfCnarxAo5MsHmgC3ehaaeBtnoMbzAKP_9TdAXGLpKadKUIottNvXSYZELjMxRSN3NK_vJhxf9gZIq21DqM"
id="tt1375666|imdb|0|0|title-maindetails" data-ga-identifier="title"
title="Users rated this 8.8/10 (520,850 votes) - click stars to rate">

Ratings: 8.8/10 from 520,850 users

Reviews: 2,179 user
552

In a world where technology exists to enter the human mind through dream invasion, a highly skilled thief is given a final chance at redemption which involves executing his toughest job to date: Inception.

<h4 class="inline">
Director: Christopher Nolan</h4>

<h4 class="inline">
Writer: Christopher Nolan</h4>

<h4 class="inline">Stars: <a href="/name/nm0000138/" itemprop="actors"
>Leonardo DiCaprio, <a href="/name/nm0330687/" itemprop="actors"
>Joseph Gordon-Levitt and <a href="/name/nm0680983/" itemprop="actors"
>Ellen Page</h4>

Structured data extracted from above snippet:
Item

Type: http://schema.org/movie
name = Jaguar Inception (2010)
aggregaterating = *Item*(1)
description = In a world where technology exists to enter the human mind through dream invasion, a highly skilled thief is given a final chance at redemption which involves executing his toughest job to...
director
text = Christopher Nolan
href = http://llrcp.ac.in/name/nm0634240/
actors
text = Leonardo DiCaprio
href = http://llrcp.ac.in/name/nm0000138/
actors
text = Joseph Gordon-Levitt
href = http://llrcp.ac.in/name/nm0330687/
actors
text = Ellen Page
href = http://llrcp.ac.in/name/nm0680983/
Item 1
Type: http://schema.org/aggregaterating
image = http://llrcp.ac.in/microformats/images/inception.jpg
ratingvalue = 8.8
bestrating = 10
ratingcount = 520,850
reviewcount = 2,179
reviewcount = 552

Above snippet illustrates the movie "Jaguar Inception" in released in year 2010 with it's description and staring. It also shows it's high rating of 8.8/10 done by more than 5 lakh users with more than 2 thousand reviews.

5.4.8 News (hNews)

hNews is a microformat for news content. hNews extends hAtom, introducing a number of fields that more completely describe a journalistic work. hNews will be one of several open standards [**62**].

hNews embedded in your website:

```
<div class="hnews hentry item">
<h4>
<a class="url entry-title" href="http://example.org/article/us-latam-obama-democracy.html" rel="bookmark">
    Obama says he wants to drive jaguar &amp;amp; not any other car</a></h4><div>
```

```
<small>&amp;nbsp;&amp;nbsp;  by
<span class="vcard"><a class="email fn"
      href="mailto:pabbijaguar@example.org">Pabbi Jaguar</a></span>,
<span class="source-org vcard"><a class="url org fn"
      href="http://www.ap.org">Associated Press</a></span>,
<a href="http://www.ap.org/newsvalues/index.html" rel="principles"><img
src="/wikidir/images/principles-button-blue.png" /></a> -
<span class="updated dtstamp"
      title="2009-04-19T18:17:29Z">19 April 2009 18:17 GMT</span>
</small>
</div>
<div class="entry-content">
<p><span class="dateline">PORT-OF-SPAIN, Trinidad</span> (AP) -- Defending his
brand
    of world politics, President Barack Obama said Sunday...</p>
<p>Both Graham and McCaskill spoke on "Fox News Sunday." Ensign was
interviewed on CNN's
    "State of the Union."</p>
</div>
<div>
<small>
    News Topics:
<a href="http://example.org/Summits" rel="tag">Summits</a>, ...,
<a href="http://example.org/Government+policy" rel="tag">Government policy</a>
<br/>
    People, Places and Companies:
<a href="http://example.org/Barack+Obama" rel="tag">Barack Obama</a>, ...,
<a href="http://example.org/Hugo+Chavez" rel="tag">Hugo Chavez</a>
</small>
</div>
<div class="geo">
<small>
    Lat: <span class="latitude">10.65715</span>
    Long: <span class="longitude">-61.483582</span>
</small>
</div>
```

Structured data extracted from above snippet:

hfeed
hentry
entry-title = Obama says he wants to drive jaguar and not any other car
entry-content = PORT-OF-SPAIN, Trinidad (AP) -- Defending his brand of world politics, President Barack Obama said Sunday... Both Graham and McCaskill spoke on "Fox News Sunday." Ensign was interviewed on...
updated = 19 April 2009 18:17 GMT
hcard
fn = Pabbi Jaguar

n
family-name = Jaguar
given-name = Pabbi
email
value = pabbijaguar@example.org
Rel
name = Obama says he wants to drive jaguar and not any other car
rel = bookmark
url = http://example.org/article/us-latam-obama-democracy.html
Rel
name = Summits
rel = tag
url = http://example.org/Summits
Rel
name = Government policy
rel = tag
url = http://example.org/Government+policy
Rel
name = Barack Obama
rel = tag
url = http://example.org/Barack+Obama
Rel
name = Hugo Chavez
rel = tag
url = http://example.org/Hugo+Chavez
Above snippet is about a demo news which says Mr. Barack Obama likes Jaguar car more than any other car. News is published by Mr. Pabbi Jaguar in year 2009.

5.4.9 Open XHTML Outlines (xoxo)

XOXO is a simple, open outline format written in standard XHTML and suitable for embedding in (X) HTML, Atom, RSS, and arbitrary XML. XOXO is one of several microformat open standards [**63**].

5.4.10 Resume (hResume)

hResume is a microformat for publishing resumes and CVs. hResume is one of several open microformat standards suitable for embedding in HTML, XHTML, Atom, RSS, and arbitrary XML [**64**].

hResume embedded in our website:

```
<div class="hResume">
<address class="vcard">
```

```
<span class="fn"><strong>Jaguar Singh</strong></span>
</address>
<address class="vcard">
<span class="adr">
<span class="street-address">Kipps Market, Sarabha Nagar</span>
</span>
</address>
<address class="vcard">
<span class="adr"><span class="locality">Ludhiana</span>, </span>
</address>
<address class="vcard">
<span class="adr"><span class="region">Punjab</span><span class="postal-code">141001</span>
</span>
</address>
<address class="vcard">
<br />
</address>
<address class="vcard">
</address>
<address class="vcard">
<span><strong>Email</strong>: <a class="email" href="mailto:jaguarsingh@example.com">jaguarsingh@gmail.com</a></span>
</address>
<address class="vcard">
<strong><span>Homepage</span></strong><span>: <a class="url" href="http://microformats.co.nr/">http://jaguarsingh.com</a></span>
</address>
<address class="vcard">
<span><strong>Phone</strong>: <span class="tel">+91-161-1234567</span></span>
</address>
<p class="summary">&amp;nbsp;
</p>
<p class="summary">I have 10 years experience with all Web 2.0 technologies– I've been working with Ajax since 1996,
 designing with pastels while others will still using tiled background images and frames... </p>
<p class="summary">&amp;nbsp;</p>
<ol class="vcalendar">
<li class="education vevent">
<div class="summary vcard">
<a class="url fn org" href="http://www.example.edu/">ABC High School</a>
<div class="adr">
<span class="locality">Civil Lines</span>,
<abbr class="region" title="Ludhiana">lud.</abbr>
</div>
</div>
```

```
  (<abbr class="dtstart" title="2001-01-24">2001</abbr> - <abbr class="dtend" title="2005-05-25">2005</abbr>)
</li>
<li class="education vevent">
<div class="summary vcard">
<a class="url fn org" href="http://www.example.edu/">XYZ Junior School</a>
<div class="adr">
<span class="locality">Tagore Nagar</span>,
<abbr class="region" title="Ludhiana">lud.</abbr>
</div>
</div>
  (<abbr class="dtstart" title="2001-01-24">2001</abbr> - <abbr class="dtend" title="2005-05-25">2005</abbr>)
</li>
</ol>
<strong>Experiance</strong>
<ol class="vcalendar">
<li class="experience vevent">
<span class="summary">President</span>,
<span class="location">ABC High School</span>,
<abbr class="dtstart" title="2008-05-01">May 2008</abbr> - <abbr title="2012-02-02">present</abbr>
</li>
<li class="experience vevent">
<span class="summary">Vice President - Computer Club</span>,
<span class="location">ABC High School</span>,
<abbr class="dtstart" title="2004-05-01">May 2004</abbr> - <abbr title="2008-05-01">May 2008</abbr>
</li>
</ol>
<strong>Job Titles</strong>
<address class="vcard">
<span class="fn n" id="pedro-name">
<span class="given-name">-Team</span>
<span class="family-name">Leader</span>
</span>
</address>
<address class="vcard">
<span class="fn n" id="pedro-name">
<span class="given-name">-Software</span>
<span class="family-name">Developer</span>
</span>
</address>
<p><br />
<strong> Skills</strong><br />
 -<a class="skill" rel="tag" href="http://example.com">C/C++</a>
<br />
```

```
 -<a class="skill" rel="tag" href="http://example.com">Java</a><br />
-<a class="skill" rel="tag" href="http://example.com">    VB</a>
<br />
<br />
<strong>Affiliations</strong><br />
<span class="affiliation vcard">
<span class="fn org">-ISTE</span><br />
<span class="fn org">-LUG</span><br />
<span class="fn org">-ACM</span><br />
   -<span class="fn org">IEEE</span><br />
</span></p>
<p>
<strong>Publications</strong><cite><br />
 -Microformats Search engine</cite>, IEEE, 2012.<br />
<cite>-Microformats Search engine</cite>, IEEE, 2012.<br />
<cite>-Microformats Search engine</cite>, IEEE, 2012.
</p>
</div>
```

Structured data extracted from the above snippet:

```
hcard
fn = Jaguar Singh
n
family-name = Singh
given-name = Jaguar

hcard
adr
street-address = Kipps Market, Sarabha Nagar
hcard
adr
locality = Ludhiana
hcard
adr
region = Punjab
postal-code = 141001
hcard
email
value = jaguarsingh@example.com
hcard
url = http://microformats.co.nr/
hcard
tel
value = +91-161-1234567
hcard
```

fn = ABC High School
adr
locality = Civil Lines
region = Ludhiana
url = http://www.example.edu/

hcalendar
date start = 2001-01-24
summary = ABC High School Civil Lines, lud.
date end = 2005-05-25
Warning: Event's start date is in the past.
hcalendar
date start = 2001-01-24
summary = XYZ Junior School Tagore Nagar, lud.
date end = 2005-05-25
Warning: Event's start date is in the past.
hcalendar
date start = 2008-05-01
summary = President
location
ABC High School
Warning: Event's start date is in the past.
hcalendar
date start = 2004-05-01
summary = Vice President - Computer Club
location
ABC High School
Warning: Event's start date is in the past.

The above snippet shows the resume of a person "Jaguar Singh" living in Ludhiana city with its email, webpage, phone and essentials for resume such as experience, job history, skills and affiliations.

5.4.11 License (rel-license)

Rel-License is a simple, open, format for indicating content licenses which is embedable in HTML or XHTML, Atom, RSS, and arbitrary XML. Rel-License is one of several microformat open standards [**65**].

Structured data extracted from the above snippet:

<a href="http://creativecommons.org/licenses/by/3.0/"
rel="license">Creative Commons Jaguar Attribution 3.0
License.

In the above snippet rel=”license” indicates that the following link is a link to a license.

5.5 Search engine view of embedded microformat

Microformats add structured tags to the web components; later search engine reads these tags to find relevant results.

For an example if we embed electronic business card “vcard microformat [**56**]” instead of plain HTML on the webpage for displaying contact information, search engines could extract structured data from the webpage and could perform semantic search on that page. Here, semantic search could be performed on the basis of say formal name (fn), family-name, first name, organization, address, email address, designation etc.

For an example a typical HTML vcard looks somewhat like this:

 Jaguar Saini test@testing.com
Kipps Market, Sarabha Nagar
Ludhiana , Punjab , 141001 India
1234567890
Software Engineer at Microsoft.

Above example is showing contact information of a software engineer at Microsoft named “Jaguar Saini” with e-mail id “test@testing.com” residing at Ludhiana, Punjab, India with phone number “1234567890”.

Above example can be written using “vcard microformat” as:

```
<div id="hcard-Jaguar-Saini" class="vcard">
<img style="float:left; margin-right:4px"
src="http://a0.twimg.com/profile_images/490760078/rs.jpg" class="photo"/>
<span class="fn">Jaguar Saini</span>
<a class="email" href="mailto:test@testing.com">test@testing.com</a>
<div class="adr">
<div class="street-address">Kipps Market, Sarabha Nagar</div>
<span class="locality">Ludhiana</span>,
<span class="region">Punjab</span>,
<span class="postal-code">141001</span>
<span class="country-name">India</span>
</div>
<div class="tel">1234567890</div>
```

Here,

It shows that the information enclosed in this division is an electronic business card.

Here id="hcard-Jaguar-Saini" is given as a name to the division and class=”vcard” shows that it’s an electronic business card.

It includes image to the vcard. Class=”vcard” shows that it’s a photograph and not anything else.

Jaguar Saini

It encloses formal name.

test@testing.com

It adds semantics to the email id by adding class=”email”. href="mailto:test@testing.com" adds hyperlink to the email id calling the default email client to send email.

It shows that the division encloses the address of the person.

Kipps Market, Sarabha Nagar

It shows that the enclosed data is the address of the person of the vcard.

Ludhiana,

class=”locality” illustrates that the enclosed data is a name of the locality.

Punjab

class=”region” illustrates that the enclosed data is a name of region.

141001

class=” postal-code” illustrates that the enclosed data is a postal code.

India

class=” country-name” illustrates that the enclosed data is a country name.

1234567890

Division encloses telephone number. class=”tel” illustrates this.

From the above microformat, search engine could extract following structured data:

hcard
fn = Jaguar Saini
n
family-name = Saini
given-name = Jaguar

org
organization-name = Microsoft
adr
street-address = Kipps Market, Sarabha Nagar
locality = Ludhiana
region = Punjab
postal-code = 141001
country-name = India
tel
value = 1234567890
email
value = test@testing.com
title = Software Engineer
photo = http://a0.twimg.com/profile_images/490760078/rs.jpg

After looking at the extracted structured data we could easily see how easy it is for the search engine to find relevant results from web.

Example – 1:

If you search for "Jaguar" who is a "person", then search engine would for the keyword "Jaguar" only and only where formal name is "Jaguar".

i.e. fn = "%Jaguar%"

Here,

(percentage sign) % = It matches zero or more characters.

Example – 2:

If you search for "Saini" which is a "family-name" then search engine would for keyword "Saini" only in the "family-name" and not at any other place.

i.e. family-name = "%Saini%"

Thus search engine would give the results where family-name="Saini" and not the first-name or street address etc.

Example – 3 (Compound Query):

We can also place compound queries.

For an example we could search for “Jaguar Saini” who is a “software engineer” at “Microsoft”.

Here, three conditions are imposed:

1. “Jaguar Saini” is a person.
2. With profession “Software Engineer”.
3. Working at “Microsoft Corporation”.

Now search engine would search for keyword “Jaguar Saini” in formal name with title as “Software Engineer” and organization as “Microsoft”.

i.e.

1. fn=”%Jaguar Saini%”
2. title=”Software Engineer”
3. organization-name = “%Microsoft%”

Here search engine would only display results satisfying all above conditions, thus getting only relevant results.

Similarly more queries could be used; search engine would search for the keywords under relevant tags finding concise but relevant results. To illustrate this process, we have created a microformats based Semantic search engine which works on a specially designed website to show the efficiency of search queries. This specially designed website is created in such a way to have many ambiguous terms thus resulting ambiguous / incorrect search results via plain pattern based search engine. However if we query the same query using microformats based search engine, it results right.

Google CSE[4] XML APIs backup our search engine to extract structured data from the webpages and to perform semantic search based on extracted structured data. Due to the limited popularity, support and similarity between microformats and microdata, in some cases microdata is being used to embed data to the webpages instead of microformats serving the same purpose. Finally efficiency of the Semantic search will be tested using various test cases specially designed for the purpose comparing the results with the other search engines such as Google and/or Bing and yahoo(powered by Bing from year 2010) etc.

[4] Custom search engine

Chapter 6
Designing a Search Engine

Microformats based search engine is being used to filter results out of the structured data extracted from the specially designed website. Microformats based search engine can be powered by almost any XML query language such as using XQuery [**66**], XML Query Language (XQL) [**67**], Yahoo Query Language (YQL) [**68**], Google CSE XML APIs [**69**] etc. Out of these and other XML query languages we thought to power our search engine with real capabilities; the capability to not only query the XHTML data but to have real techniques and algorithms being used by current search engines comparing final results with some other search engines such as Google, Yahoo/Bing etc. To fulfill such desires, we were left to use the search engine APIs instead of a simple query language. Search narrowed down to Yahoo Searchmonkey [**70**] and Google CSE XML APIs [**69**] fulfilling above said requirements. Finally we choose Google CSE XML APIs to power our search engine considering the performance, availability and ease of use.

In order to use Google CSE XML APIs extracting structured data from webpages; we need to submit our website for indexing at Google webmasters [**71**]. Google account [**72**] is required to sign in to Google Webmasters. After Website is being indexed, custom search engine is being created to query the extracted structured data.

Note: Google web crawlers may take up to 4-6days to index a website.

6.1 Creating Google Custom Search Engine (CSE)

Steps to be taken to submit a website for indexing followed by creating a Google CSE:

1. Sign in to Google Webmasters using Google Account.
2. After sign in; click on “Add a site” (Figure 6.1) button on the top right corner of the screen to submit website to Google for indexing.

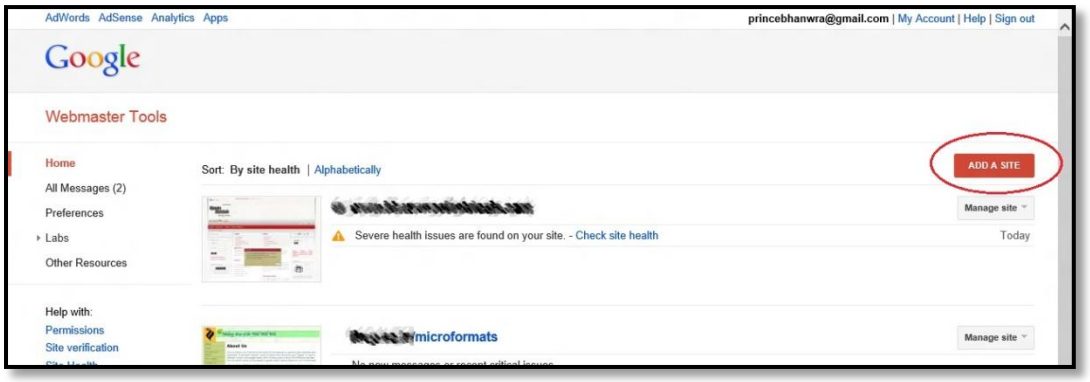

Figure 6.1: Google Webmasters home screen

3. Enter site domain for indexing (Figure 6.2).

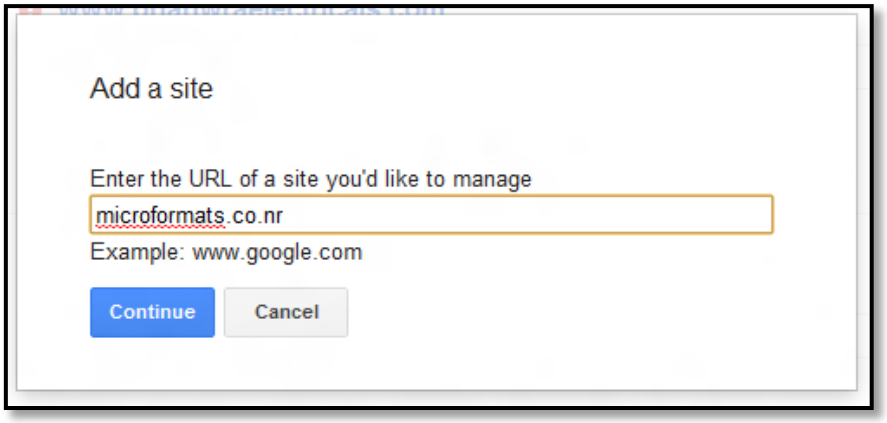

Figure 6.2: Submitting domain name for indexing

4. Verify the ownership of the site by uploading the unique file created by Google on the root of webserver; then click "Verify" button (Figure 6.3).
5. Successful verification would lead to the Google Website Dashboard (Figure 6.4); now click "Custom Search" under "Labs".
6. Now it would open a wizard/webpage asking for the basic information about your search engine such as name of the search engine, language, keywords, site to search enable/disable auto completion etc. and click "Save and get code" button at the bottom of the webpage (Figure 6.5).

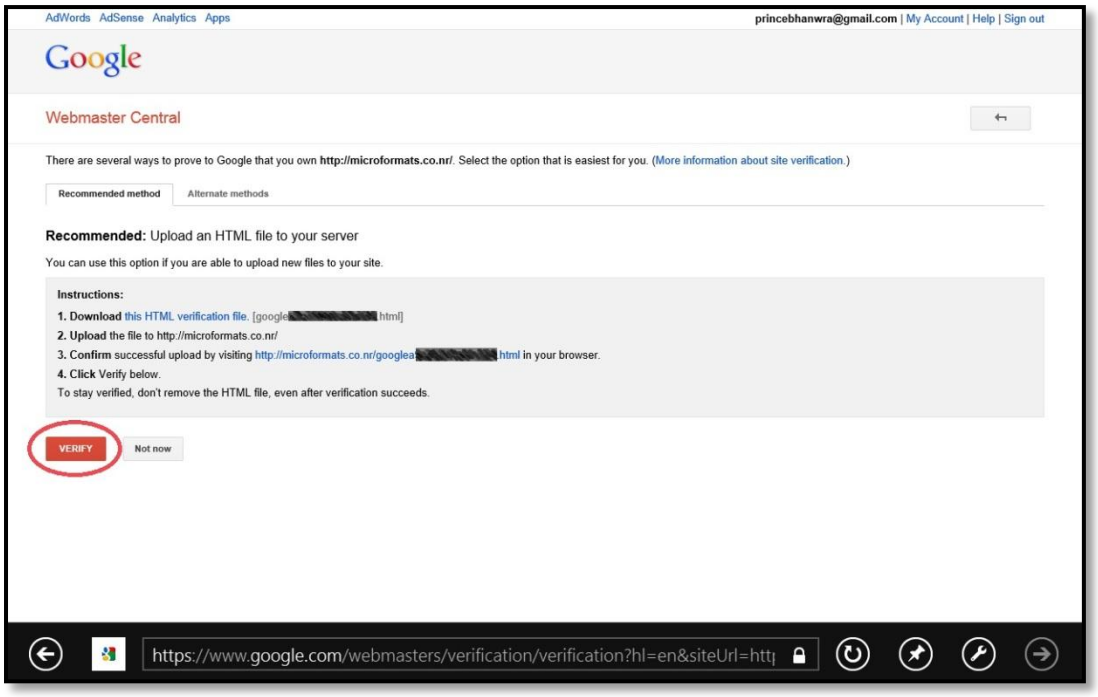

Figure 6.3: Site Ownership Verification

7. Now copy-paste the retrieved code on the page to add search engine to the page.

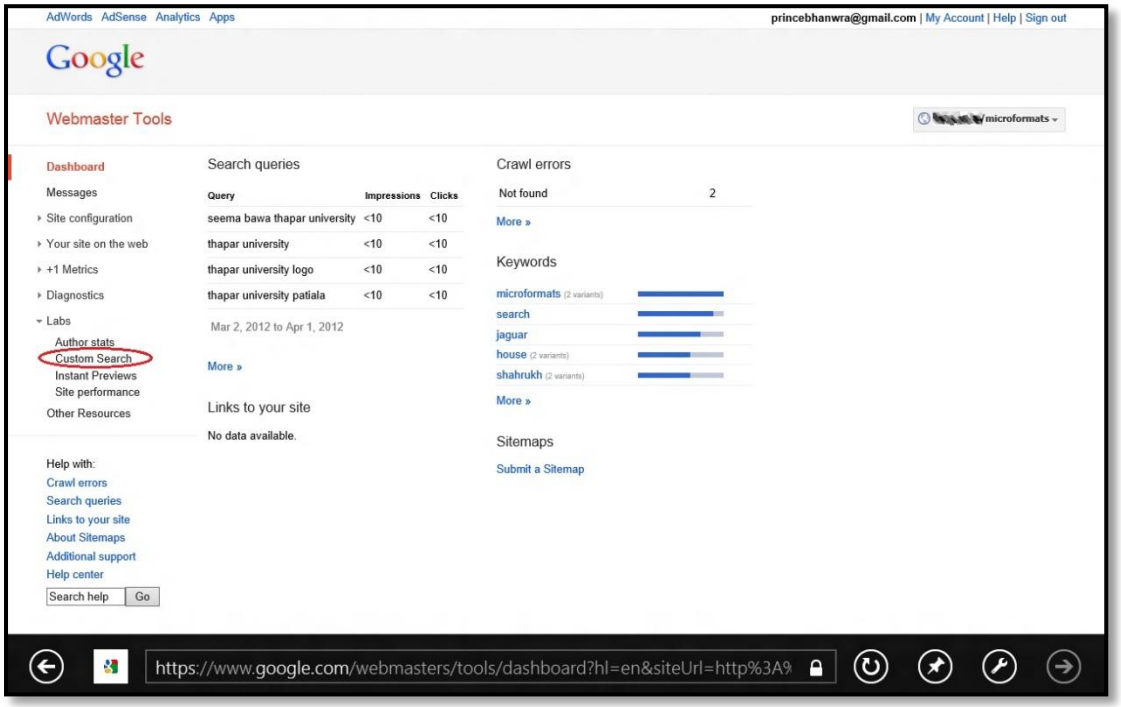

Figure 6.4: Site Dashboard

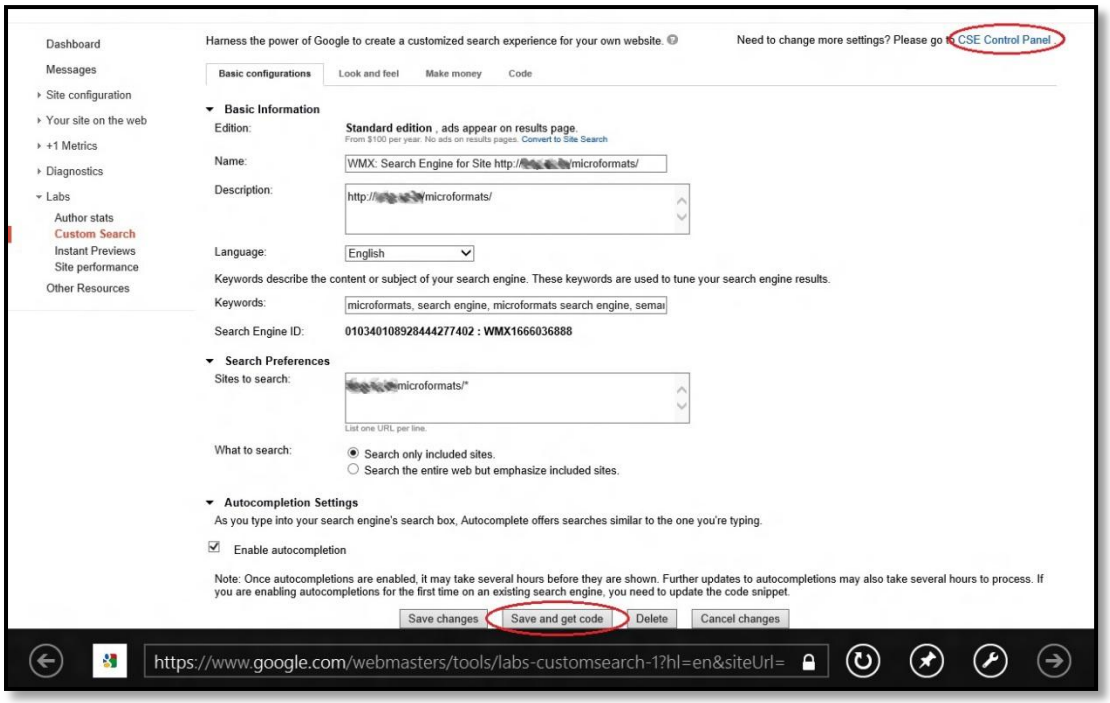

Figure 6.5: Creating Custom Search Engine

6.2 Configuring Google Custom Search Engine (CSE) for Structured Queries

After custom search engine is created, we need to configure our CSE to perform structured queries on the extracted structured data. To perform the above said task in Google CSE, structured data must be embedded into the webpages in the form of PageMaps [**73**], Microformats [**4**], RDFa [**1**], Microdata [**2**], META Tags [**74**] or Page Date [**75**]. However, Google Search does not use PageMaps or META tags while generating rich snippets. Information such as microformats, microdata, RDFa and the page date are used while generating snippet [**76**].

As reason stated in section1.1, we have been using Microformats or rarely Microdata (based on Microformats) to embed semantics to the webpages. Then the Google extracts structured data from Microformats to be used in Rich Snippets [**77**] i.e. extended presentations of standard Google search results. A subset of this data is available for use in Custom Search's structured data operators. In case of embedding custom data we are interested to enter does not correspond to a defined microformat;

PageMaps are used, as google creates PageMaps from the extracted structured data and PageMap query is used to perform semantic search on extracted structured data. Extracted data could be seen using Rich Snippets Preview Tool in Webmaster tools [**77**].

For an example if we have marked up our page with the Microformat hrecipe standard, we can perform number of semantic operations on it such as sorting on the basis of star rating with an operator like *&sort=recipe-ratingstars* [**76**].

Semantic search operations that can be performed over structured data are [**78**]:

- Filter by Attribute
    - Results with a specific attached DataObject, such as a review
    - Results with a DataObject with a given field, such as a review with a price range.
    - Results with a specific value of a field, such as a review with 5 stars.

- Sort by Attribute
- Bias by Attribute
- Restrict to Range

Detailed example of embedding business card using microformats:

```
<div class="vcard">
<p><strong class="fn">Prince Bhanwra</strong></p>
<p><span class="title">Student</span> at <span class="org">Thapar University</span></p>
<p><span class="adr">
<span class="street-address">P.O. Box 32</span>,
<span class="locality">Bhupindra Road</span>,
<span class="locality">Patiala</span>,
<span class="region">India</span>
<span class="postcode">147004</span>
</span></p>
</div>
```

Extracted rich snippet from the page containing above microfromat:

hcard

fn = Prince Bhanwra
n
family-name = Bhanwra
given-name = Prince
org
organization-name = Thapar University
adr
street-address = P.O. Box 32
locality = Bhupindra Road
region = India
title = Student

Structured data queries can be used to filter, sort, bias or restrict search results in Custom Search are:

more:pagemap:adr
more:pagemap:adr-locality:bhupindra
more:pagemap:adr-region:india
more:pagemap:adr-street_address:32
more:pagemap:hcard-adr
more:pagemap:hcard-fn:prince_bhanwra
more:pagemap:hcard-name
more:pagemap:hcard-org
more:pagemap:hcard-title:student
more:pagemap:name
more:pagemap:name-family_name:bhanwra
more:pagemap:name-given_name:prince
more:pagemap:org-organization_name:thapar_university
more:pagemap:person-location:bhupindra_road
more:pagemap:person-location:india
more:pagemap:person-org:thapar
more:pagemap:person-role:student

Note: Complete set of possible queries on the above example is available online at [**79**].
Syntax to filter query in search [**76**]:

more:pagemap:TYPE-NAME:VALUE

Example:

Thapar University more:pagemap:hcard-fn:prince

Here,

more: operator is for refinement labels telling Custom Search that we are performing a structured query.

Pagemap: It tells to refine the results on the basis of specific attributes in the indexed PageMaps.

hcard-fn: The remaining elements of operator specify the content the restriction drills down into.

In our case it tells that we are searching for the hcard where formatted name (fn) of the person is "%Prince%". And Thapar University written at the start specifies that the page should also contain the keyword Thapar University.

Note: % specifies zero or more characters

Syntax to filter query in search [**76**]:

more:pagemap:*TYPE-NAME*:*VALUE*

Example:

Thapar University more:pagemap:hcard-fn:prince

Here,

more: operator is for refinement labels telling Custom Search that we are performing a structured query.

Pagemap: It tells to refine the results on the basis of specific attributes in the indexed PageMaps.

hcard-fn: The remaining elements of operator specify the content the restriction drills down into.

In our case it tells that we are searching for the hcard where formatted name (fn) of the person is "%Prince%". And Thapar University written at the start specifies that the page should also contain the keywords Thapar University.

Note: % specifies zero or more characters

For the ease of not remembering the filter operators and to make them available for a regular user above said query and similar queries can be saved in the labels called Google Refinements. Refinements are labels that can be applied to Google custom search engine index to users more easily find the information they are looking for [**80**].

Steps to create a Refinement label in Google custom search engine (CSE):

1. On the Custom Search Engine page; click "CSE Control Panel" button on the top right corner of the window as shown in Figure 6.5.
2. On CSE Control Panel; click on "Refinements" on the left menu bar as shown in Figure 6.6.
3. It would list all the Refinements; if already created. Click on "Add Refinement" button to add new Refinement as shown in Figure 6.6.
4. In the Refinement name box, type a label.
5. Specify any other options as per need and click "Save button" as shown in Figure 6.6.

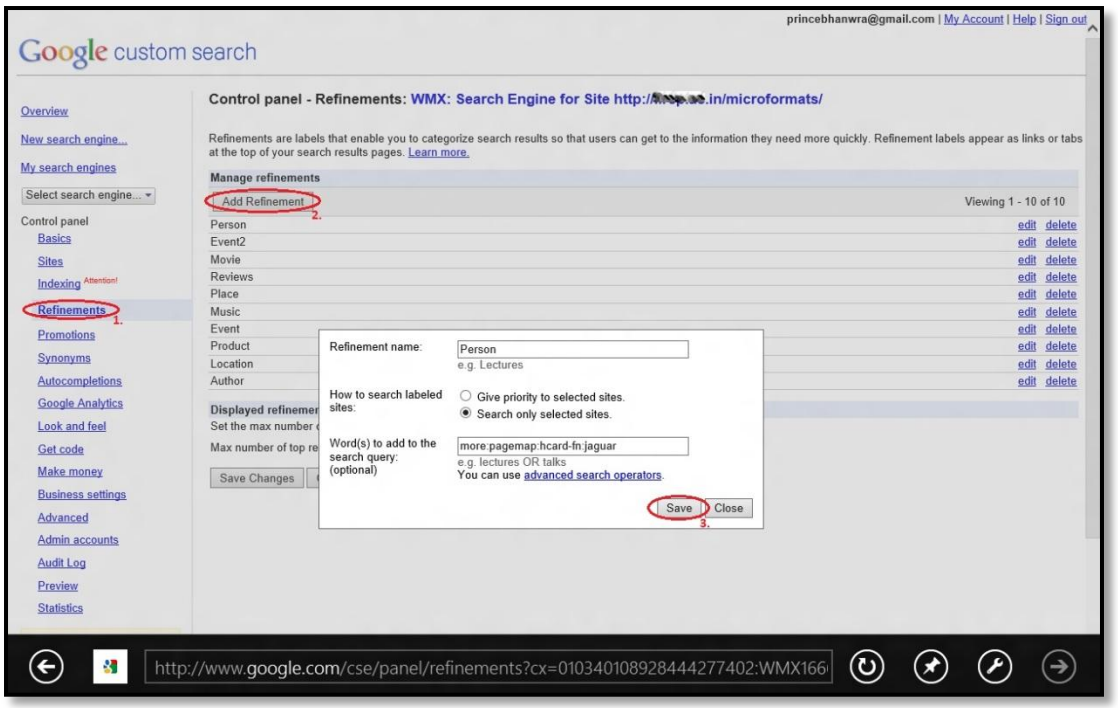

Figure 6.6: Adding Refinements in CSE Control Panel

6. It will prompt to add sites to the labels. Click “Go to Sites tab” button to add sites to labels right now or click “Close” button to do it later and continue as shown in Figure 6.7. Steps to add sites to labels are shown below.
7. Repeat the above written steps for every label we are interested to create.
8. Now, Under Displayed refinements, select the number of refinements to be displayed in search results page.
9. Finally click “Save Changes” button.

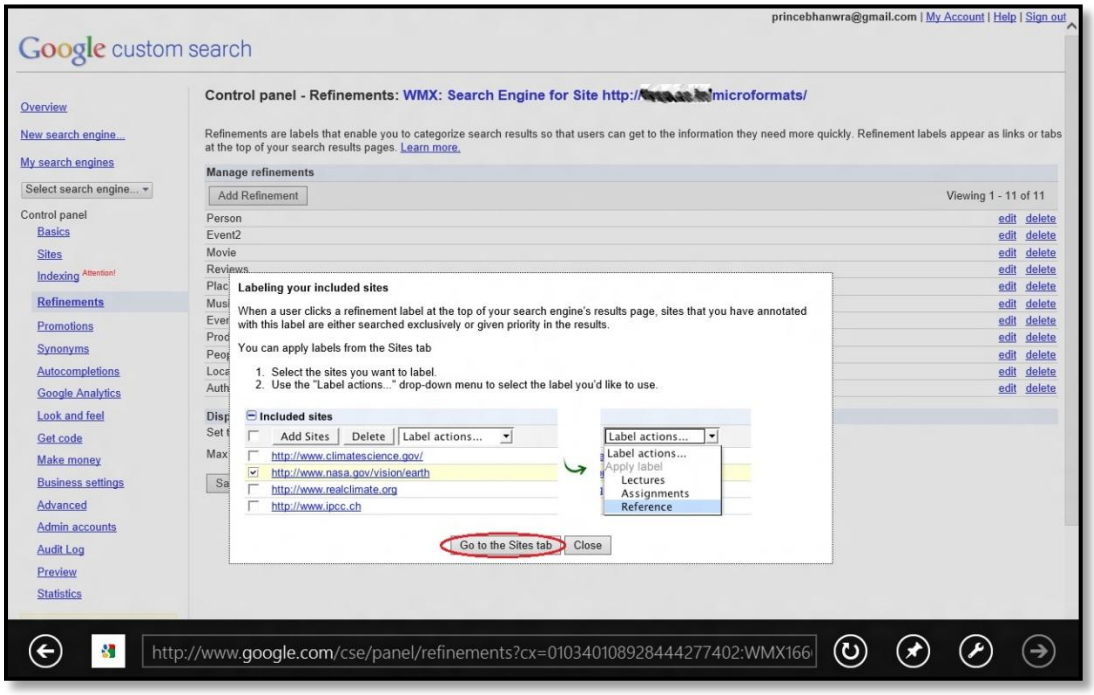

Figure 6.7: Warning to Assign Sites to Refinement Label

Steps to apply labels to site(s):

1. Click on “Sites” link on the left-hand menu in CSE Control Panel or you might have reached there if you have clicked “Go to Sites Tab” button as illustrated in step 6 above.
2. Now select a site and then click Label actions as shown in Figure 6.8.
3. Now select the label we want to add. In our case “Person” label as shown in Figure 6.8.
4. Repeat the above said steps for all the sites you want to label.

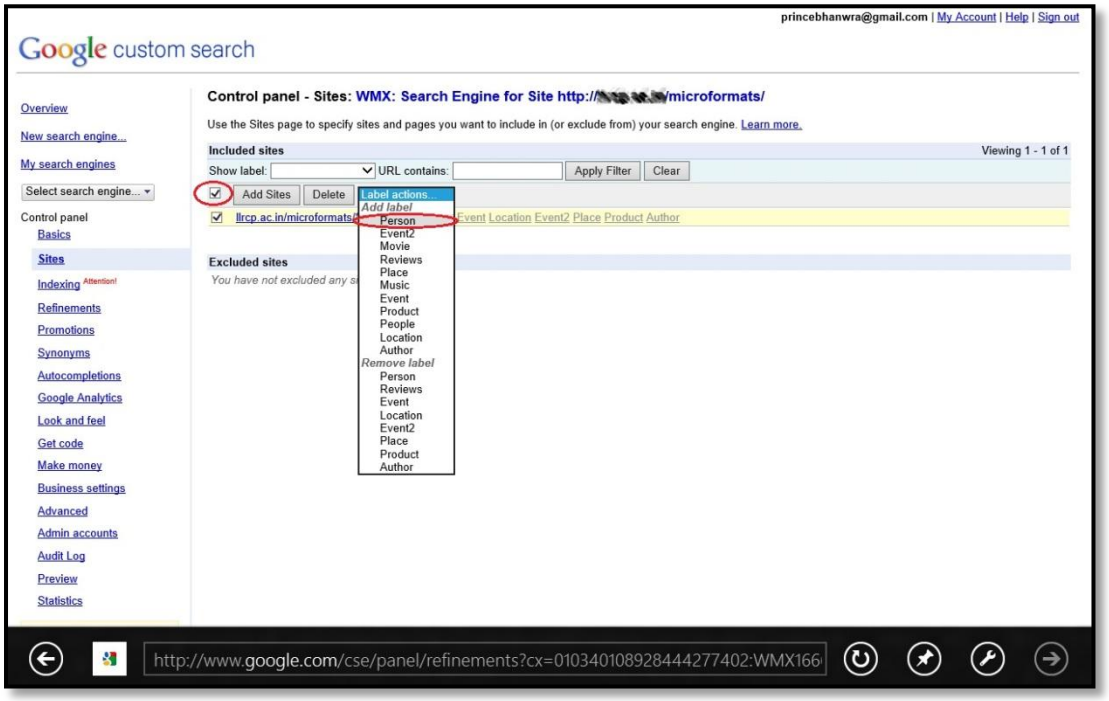

Figure 6.8: Adding Sites to Refinement Label

Chapter 7

Results and Observation

Results of the implemented search engine should be compared with normal pattern based search engine for authenticity. We would be comparing the result of our semantic search engine with Google and/or Yahoo/Bing search engine. For unbiased and accurate results, search engines to be compared should only work on our specially designed website i.e. www.microformats.co.nr. In order to perform search on a particular domain "site:" operator is being used in Google, Yahoo/Bing search engine.

Example:

Jaguar site:microformats.co.nr

Here, keyword "Jaguar" would be searched only in domain "www.microformats.co.nr".

Or an alternative way of performing the same job is by creating Custom Search Engine (CSE) in Google (as shown above), BOSS (Build your own search service) in Yahoo/Bing.

In our case we have built a custom search engine as shown in section 6.1. Here Google search request is a standard HTTP GET command. It includes a collection of parameters relevant to our queries. These parameters are included in the request URL as name=value pairs separated by ampersand (&) characters. Parameters include data like the search query and a unique CSE ID (cx) that identifies the CSE that is making the HTTP request. The WebSearch or Image Search service returns XML results in response to HTTP requests [**69**].

For an example:

http://www.google.com/cse/home?cx=010340108928444277402:WMX1666036888

Here by passing our custom search engine's unique ID as value to the variable cx redirects to our Microformats based custom search engine instead of Google public search engine as shown in Figure. Please note it's the same ID generated by Google as shown in Figure 6.5.

Or we could simply add the code generated in step 7 under section 6.1 (Creating Google Custom Search Engine (CSE)) to the webpage where we want our custom search engine.

An implemented version of above said can be found at webpage address "microformats.co.nr/search.html" and "microformats.co.nr/plainsearch.html".

Other parameters which could be passed to the URL are [**69**]:

1. Enable/Disable Simplified and Traditional Chinese Search. Please note in all the cases value 0 indicates disabled and 1 means enabled option.
2. Enable/Disable country based search.
3. Automatic filtering the Google search results.
4. Boosting country based search results.

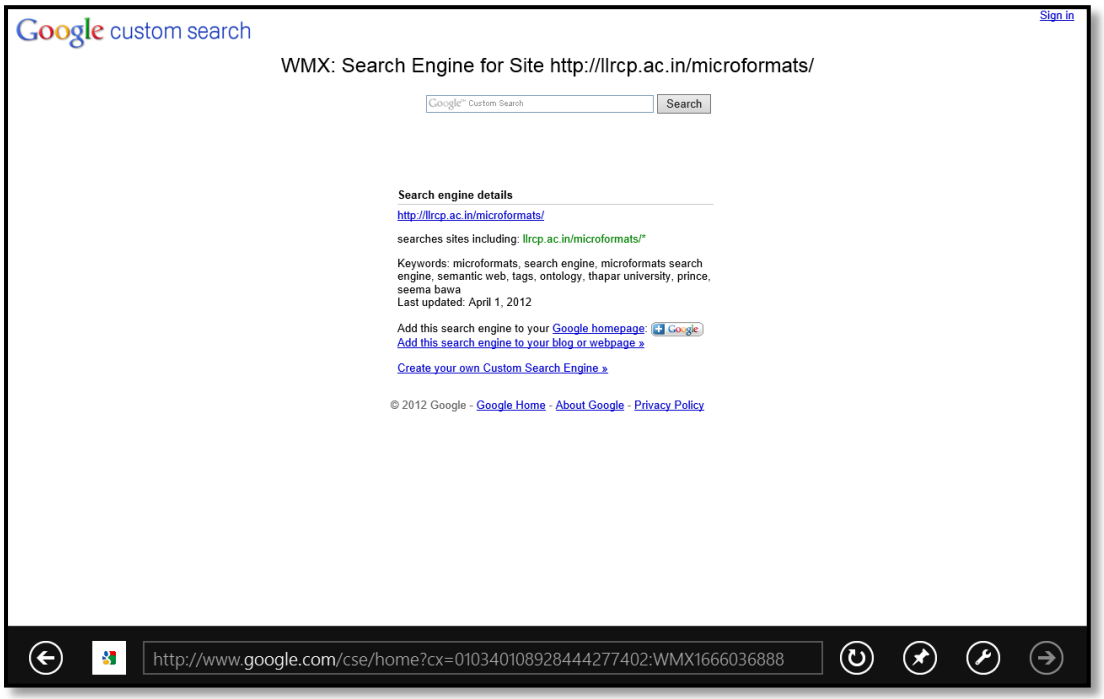

Figure 7.1: Microformats based CSE

5. Performing compound queries, using logical operators.
6. Setting character encoding scheme.
7. Languages restrict results.
8. Number of search results to return.

9. Setting output method say JSON [**81**] or XML.
10. Adding query keywords.
11. Enable/Disable safe search.
12. Fixing starting result index.
13. Sorting the search results etc.

Note: Some of the above said and other parameterized feature(s) is/are available under Google Site Search and not Google CSE (Custom Search). Google Site Search is an extended paid Google CSE service with some useful extra features. More differences between Google Site Search and Google CSE are available at reference [**82**]. Pricing details of Google site search is available at reference [**83**].

In our case we would be using the webpage with embedded with Google CSE code, generated in step 7at section 6.1 (Creating Google Custom Search Engine (CSE)). Here simple (non-structured) queries would be performed simply by typing query in the custom search engine whereas structured query is being performed by using different predefined Refinements tabs.

For an example:

If we search for the keyword "Jaguar" in the custom search engine without using refinements, it displays approximately 21 results split into 3 pages as shown in Figure 7.2.

Whereas using "Person" refinement created in step displays only 3 relevant results as shown in figure Figure 7.3 wherein:

1. First result displays the information for a person name Pabbi Jaguar.
    a. Notice It being the result of a person, It shows rich snippet showing name, email address, complete address, designation, organization and the phone number then and there in the result.
2. Second result shows the residential information of the person "Jaguar Singh".
3. Last result shows a review done by the person named "Jaguar Singh".

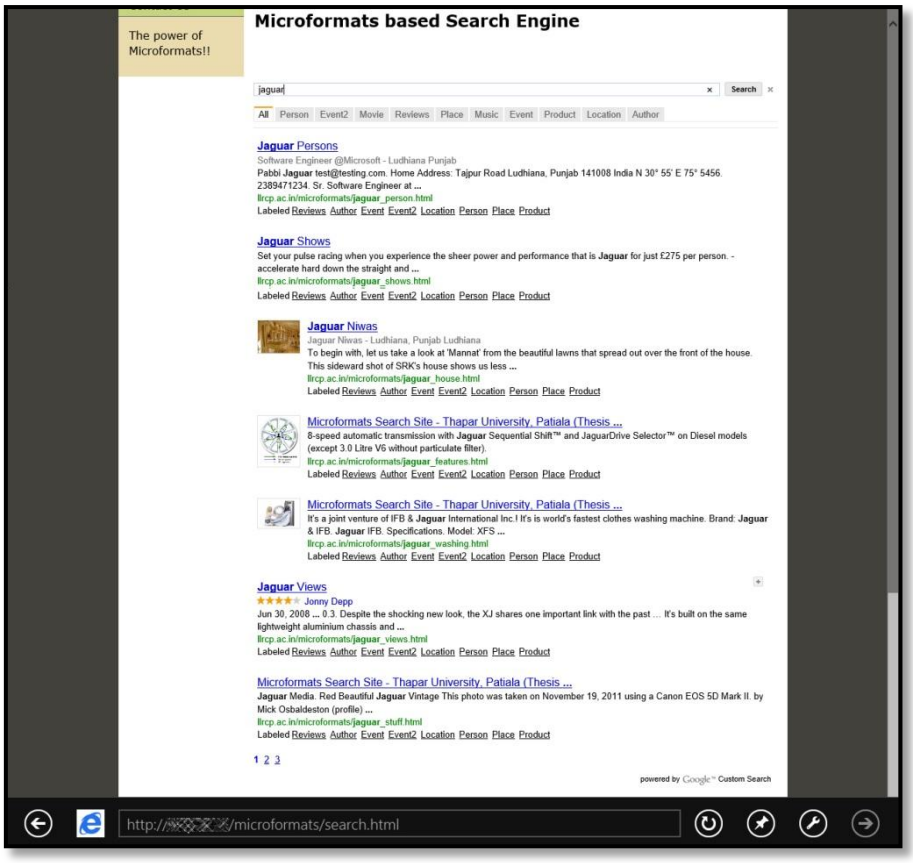

Figure 7.2: Search results without Refinement (Structured data)

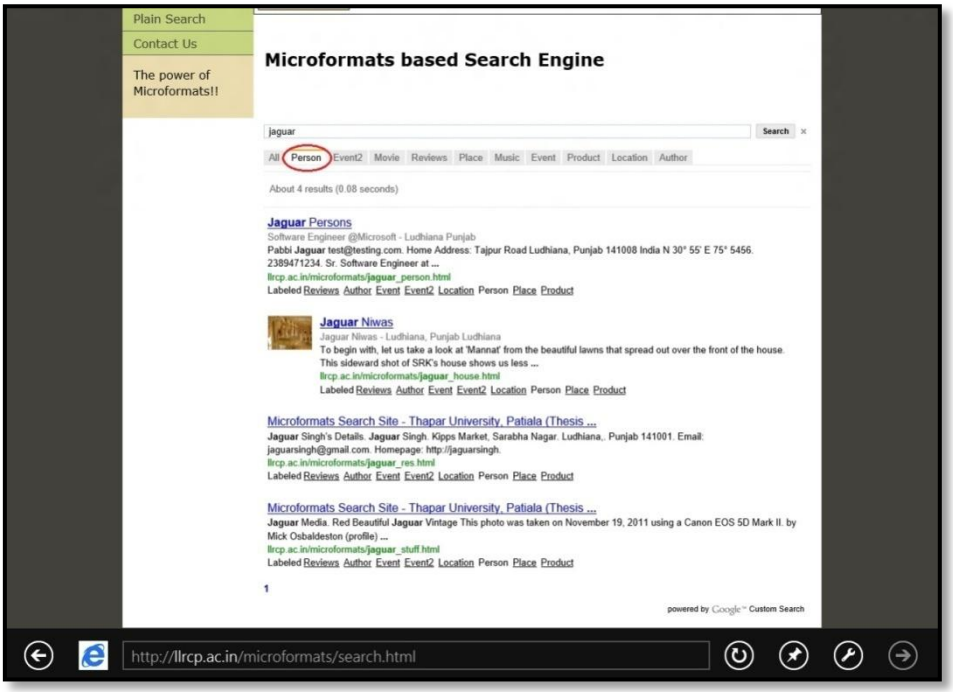

Figure 7.3: Search Results with Refinements (Structured Data)

More such queries can be performed at “microformats.co.nr/search.html” and “microformats.co.nr/plainsearch.html” selecting different refinements tabs.

Other possible, not exhaustive search terms which may be used to extract search results in the above scenario with their accuracy are shown in TABLE II.

Accuracy is calculated by F-score (F-measure) at. It is a measure of a test’s accuracy. It considers both precision (p) and recall (r) [47] of the test to compute final accuracy. Precision is seen as a measure of exactness or quality, whereas recall is a measure of completeness or quantity.

$$precision(p) = \frac{\left|\left\{\begin{matrix} retrievedrelevant \\ results \end{matrix}\right\} \cap \left\{\begin{matrix} retrieved \\ results \end{matrix}\right\}\right|}{\left|\left\{\begin{matrix} retrieved \\ results \end{matrix}\right\}\right|}$$

$$recall(r) = \frac{\left|\left\{\begin{matrix} retrievedrelevant \\ results \end{matrix}\right\} \cap \left\{\begin{matrix} retrieved \\ results \end{matrix}\right\}\right|}{|\{totalrelevantresults\}|}$$

In search engines, a perfect precision score of 1.0 means that every result retrieved by a search was relevant (but says nothing about whether all relevant documents were retrieved) whereas a perfect recall score of 1.0 means that all relevant documents were retrieved by the search (but says nothing about how many irrelevant documents were also retrieved). Thus F-measure combines precision and recall by taking harmonic mean of them. Note precision can also be evaluated at a given cut-off rank, considering only the topmost results returned by the system. This measure is called precision at n or P@n.

$$F = 2 \times \frac{precision \times recall}{precision + recall}$$

Example:

Query “Jaguar Person” returns 16 results out of which 2 are relevant results and it does not miss any relevant result. Then,

$$precision = \frac{|\{\#2\} \cap \{\#16\}|}{|\{\#16\}|} = \frac{2}{16} = 0.125$$

$$recall = \frac{|\{\#2\} \cap \{\#16\}|}{|\{\#2\}|} = \frac{2}{2} = 1.0$$

$$F = 2 \times \frac{0.125 \times 1.0}{0.125 + 1.0} = 0.222$$

Details of the above query are shown at serial no. 2 in TABLE II. Snapshot of the above search term "Jaguar Person" with regular search results is shown in Figure 7.4.

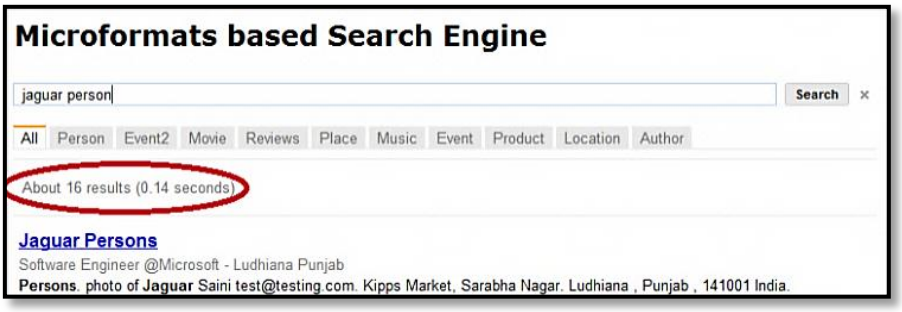

Figure 7.4: Search Results count with keyword "Jaguar Person" on regular search engine.

Table 7.1: Accuracy of different search terms.

Search Keyword(s)	**Total relevant results**	**Retrieved results**	**Retrieved relevant results**	**Precision (p)**	**Recall (r)**	**F-measure**	**Accuracy (%age)**
Jaguar	2	20	2	$^2/_{20} = 0.1$	$^2/_2 = 1.0$	$2 \times \frac{0.1 \times 1.0}{0.1 + 1.0} = 0.181$	$0.181 \times 100 = 18.1\%$
Jaguar Person	2	16	2	$^2/_{16} = 0.125$	$^2/_2 = 1.0$	$2 \times \frac{0.125 \times 1.0}{0.125 + 1.0} = 0.222$	$0.222 \times 100 = 22.2\%$
Jaguar Person -Car	2	12	2	$^2/_{12} = 0.166$	$^2/_2 = 1.0$	$2 \times \frac{0.166 \times 1.0}{0.166 + 1.0} = 0.285$	$0.285 \times 100 = 28.5\%$

Search Keyword(s)	Total relevant results	Retrieved results	Retrieved relevant results	Precision (p)	Recall (r)	F-measure	Accuracy (%age)
Jaguar microsoft*	2	5	2	$^2/_5 = 0.4$	$^2/_2 = 1.0$	$2 \times \frac{0.4 \times 1.0}{0.4 + 1.0} = 0.571$	$0.571 \times 100 = 57.1\%$
Jaguar microsoft – car*	2	3	1	$^1/_3 = 0.333$	$^1/_2 = 0.5$	$2 \times \frac{0.333 \times 0.5}{0.333 + 0.5} = 0.398$	$0.398 \times 100 = 39.8\%$
Jaguar Punjab*	2	3	2	$^2/_3 = 0.666$	$^2/_2 = 1.0$	$2 \times \frac{0.666 \times 1.0}{0.666 + 1.0} = 0.799$	$0.799 \times 100 = 79.9\%$
Jaguar Engineer*	2	1	1	$^1/_1 = 1.0$	$^1/_2 = 0.5$	$2 \times \frac{1.0 \times 0.5}{1.0 + 0.5} = 0.666$	$0.666 \times 100 = 66.6\%$
Jaguar Software*	2	3	2	$^2/_3 = 0.666$	$^2/_2 = 1.0$	$2 \times \frac{0.666 \times 1.0}{0.666 + 1.0} = 0.799$	$0.799 \times 100 = 79.9\%$
Jaguar Software Engineer*	2	1	1	$^1/_1 = 1.0$	$^1/_2 = 0.5$	$2 \times \frac{1.0 \times 0.5}{1.0 + 0.5} = 0.666$	$0.666 \times 100 = 66.6\%$
Jaguar Software Developer*	2	1	1	$^1/_1 = 1.0$	$^1/_2 = 0.5$	$2 \times \frac{1.0 \times 0.5}{1.0 + 0.5} = 0.666$	$0.666 \times 100 = 66.6\%$

*Queries can only be performed if additional information about the person (subject) in query is known.

Chapter 8
Conclusion

Flexible nature of the Microformats intended to solve simpler problems adapting the current behaviours and patters encouraged us to show the power of Microformats in search engine. To perform the above said task a website is specially designed in which different Microformats were embedded instead of simple HTML and many ambiguous terms were used on different web pages making it a challenging task for the search engine to extract the right information from the webpages simulating a live issues faced by the search engines in huge web. Then a custom search engine is being created to show the search results based on the structured data extracted from the Microformats webpages. Custom search engine is back up by Google XML APIs otherwise it could be powered up by almost any XML query language such as XQuery, XML Query Language, Yahoo Query Language (YQL) etc. Idea was to use real techniques and algorithms so as to explore the real capabilities of the search engine showing results not only by querying the XML data but to use the real capabilities and algorithms of the search engine.

Finally results are compared live with other search engines such as Google and/or Yahoo/Bing. Efficiency of the search results makes demonstrates the power of Microformats in search engines. Above said work can also be performed via other Invisible semantic languages such as RDF, however we choose Microformats because of the reason stated in section 1.1 (Why Microformats).

Efficiency of the search results could even be increased by using personalized search results on the basis microformats creating a user profile. Structure of the profile could be as illustrated at reference [**84**]. Further the same work could be done efficiently using YQL implemented via Yahoo BOSS (Build Your Own Search Service) [**85**] and Yahoo Searchmonkey [**70**].

References

[1] (2012, February) RDF Primer. [Online]. http://www.w3.org/TR/rdf-primer/

[2] (2012, February) Metadata at W3C. [Online]. http://www.w3.org/Metadata/

[3] (2012, February) Introduction to OWL. [Online]. http://www.w3schools.com/rdf/rdf_owl.asp

[4] (2012, February) Microformats | About Microformats. [Online]. http://microformats.org/about

[5] (2012, February) XHTML 1.0: The Extensible HyperText Markup Language (Second Edition). [Online]. http://www.w3.org/TR/xhtml1/

[6] (2012, February) Microformats | Code & Tools. [Online]. http://microformats.org/code-tools

[7] (2012, March) OWL Web Ontology Language. [Online]. http://www.w3.org/TR/2004/REC-owl-features-20040210/

[8] (2012, February) Welcome to the microformats wiki! • Microformats Wiki. [Online]. http://microformats.org/wiki/

[9] (2012, March) poshformats · Microformats Wiki. [Online]. http://microformats.org/wiki/poshformats

[10] (2012, March) microdata · Microformats Wiki. [Online]. http://microformats.org/wiki/microdata

[11] (2012, March) 5 Microdata — HTML Standard. [Online]. http://www.whatwg.org/specs/web-apps/current-work/multipage/microdata.html

[12] S.A. McIlraith, T.C. Son, and Zeng Honglei, "Semantic Web services," *IEEE*, vol. 16, no. 2, April 2005.

[13] Yong-Ju Lee and Chang-Su Kim, "Building Semantic Ontologies for RESTful Web Services," in *IEEE International Conference*, 2010, pp. 383 - 386.

[14] S.A. McIlraith, T.C. Son, and Honglei Zeng, "Semantic Web services," *Intelligent Systems, IEEE*, vol. 16, no. 2, pp. 46 - 53, April 2005.

[15] Ossama H. Embarak, "Integration Of Users Preferences And Semantic Structure To Solve The Cold Start Problem," *IEEE*, pp. 244 - 249, Jun 2011.

[16] A.S. Cernian, D. Carstoiu, A.M. Vladu, A. Olteanu, V. Sgarciu M. Mohirta, "A Semantic Web Based Scientific News Aggregator," *IEEE*, pp. 285 - 289, May 2011.

[17] C. Pedrinaci et al., "Semantic Business Process Management: Scaling up the Management of Business Processes," in *IEEE International Conference*, 2008, pp. 546 - 553.

[18] Jia Zhang et al., "Toward Semantics Empowered Biomedical Web Services," in *IEEE International Conference*, 2011, pp. 371 - 378.

[19] "Web Semantics in the Clouds," *IEEE*, vol. 23, no. 5, pp. 82 - 87, October 2008.

[20] Chaoqing Lv, T. Kobayashi, K. Agusa, Kun Wu, and Qing Zhu, "Image Semantic Search Engine," in *IEEE International Conference*, 2009, pp. 156 - 159.

[21] J.M. Kassim and M. Rahmany, "Introduction to Semantic Search Engine," in *IEEE International Conference*, 2009, pp. 380 - 386.

[22] (2011, March) hakia Inc., Corporate Page. [Online]. http://company.hakia.com/about.html

[23] (2011, March) SenseBot - semantic search engine that finds sense on the Web. [Online]. http://www.sensebot.net/

[24] (March, 2012) How It Works | DeepDyve. [Online]. http://www.deepdyve.com/how-it-works

[25] Ph.D., Chief Technical Officer Dan Albro, Ph.D., Chief Scientist Kathleen Dahlgren, "Cognition Technology Resources Overview Semantic Map, System Architecture and Tools," Cognition Technologies, Inc., White Paper 2008.

[26] Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal C Doshi, and Joel Sachs Li Ding, "Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management," in *ACM Press*, 2004.

[27] (2012, March) posh · Microformats Wiki. [Online]. http://microformats.org/wiki/posh

[28] (2012, March) hCalendar 1.0 · Microformats Wiki. [Online]. http://microformats.org/wiki/hcalendar

[29] (2012, March) ISO - FAQs - Date and time format. [Online]. iso.org/iso/date_and_time_format

[30] (2012, March) HTML abbr tag. [Online]. w3schools.com/tags/tag_abbr.asp

[31] Tantek Çelik, Bud Gibson, Ryan King, and Eron Wright. (2012, March) location-formats · Microformats Wiki. [Online]. microformats.org/wiki/location-formats

[32] Tantek Çelik. (2012, March) Geo · Microformats Wiki. [Online]. http://microformats.org/wiki/geo

[33] Tantek Çelik. (2012, March) adr · Microformats Wiki. [Online]. http://microformats.org/wiki/adr

[34] (2012, March) HTML & CSS - W3C. [Online]. http://www.w3.org/standards/webdesign/htmlcss

[35] (2012, March) JavaScript Web APIs - W3C. [Online]. http://www.w3.org/standards/webdesign/script

[36] (2012, March) Adobe Photoshop CS5, our latest picture and image editor software. [Online]. http://www.adobe.com/products/photoshop.html

[37] (2012, March) Web design software, HTML editor | Adobe Dreamweaver CS5.5. [Online]. http://www.adobe.com/products/dreamweaver.html

[38] (2012, March) FileZilla - The free FTP solution. [Online]. http://filezilla-project.org/

[39] (2012, March) Welcome! - The Apache HTTP Server Project. [Online]. http://httpd.apache.org/

[40] (2012, March) HTML doctype declaration. [Online]. http://www.w3schools.com/tags/tag_doctype.asp

[41] (2012, March) HTML head tag. [Online]. http://www.w3schools.com/tags/tag_head.asp

[42] (2012, March) HTML title tag. [Online]. http://www.w3schools.com/tags/tag_title.asp

[43] (2012, March) HTML body tag. [Online]. http://www.w3schools.com/tags/tag_body.asp

[44] (2012, March) HTML5 b Tag. [Online]. http://www.w3schools.com/html5/tag_b.asp

[45] (2012, March) Cascading Style Sheets. [Online]. http://www.w3.org/Style/CSS/Overview.en.html

[46] (2012, March) CSS Introduction. [Online]. http://www.w3schools.com/css/css_intro.asp

[47] (2012, March) JavaScript Introduction. [Online]. http://www.w3schools.com/js/js_intro.asp

[48] (2012, March) Enable JavaScript in my browser - AdSense Help. [Online]. http://support.google.com/adsense/bin/answer.py?hl=en&answer=12654

[49] (2012, March) How to enable JavaScript in your browser and why. [Online]. http://enable-javascript.com/

[50] (2012, March) The GNU General Public License v3.0 - GNU Project - Free Software Foundation (FSF). [Online]. http://www.gnu.org/copyleft/gpl.html

[51] (2012, March) compound-microformat · Microformats Wiki. [Online]. http://microformats.org/wiki/compound-microformat

[52] (2012, March) Operator : Add-ons for Firefox. [Online]. https://addons.mozilla.org/en-US/firefox/addon/operator/

[53] (2012, March) Oomph - A Microformats Toolkit. [Online]. http://oomph.codeplex.com/

[54] (2012, March) Zappatic.net. [Online]. http://zappatic.net/projects/safarimicroformats

[55] (2012, March) Chrome Web Store - Microformats for Google Chrome™. [Online]. https://chrome.google.com/webstore/detail/oalbifknmclbnmjlljdemhjjlkmppjjl?hl=en-US

[56] Tantek Çelik and Brian Suda. (2012, March) hCard 1.0 · Microformats Wiki. [Online]. http://microformats.org/wiki/hcard

[57] (2012, March) hReview 0.4 (in progress) · Microformats Wiki. [Online]. http://microformats.org/wiki/hreview

[58] Paul Lee and Tantek Çelik. (2012, March) hProduct · Microformats Wiki. [Online]. http://microformats.org/wiki/hproduct

[59] Kevin Marks and Tantek Çelik. (2012, March) Vote Links · Microformats Wiki. [Online]. http://microformats.org/wiki/vote-links

[60] Ian Hickson and Google Inc. (2012, March) HTML Microdata. [Online]. http://www.w3.org/TR/microdata/

[61] Manu Sporny and Martin McEvoy. (2012, March) hAudio 0.9.1 · Microformats Wiki. [Online]. http://microformats.org/wiki/haudio

[62] Jonathan Malek. (2012, March) hNews 0.1 · Microformats Wiki. [Online]. http://microformats.org/wiki/hnews

[63] Tantek Çelik. (2012, March) XOXO 1.0: Extensible Open XHTML Outlines · Microformats Wiki. [Online]. http://microformats.org/wiki/xoxo

[64] Ryan King. (2012, March) hResume · Microformats Wiki. [Online]. http://microformats.org/wiki/hresume

[65] Tantek Çelik. (2012, March) rel="license" · Microformats Wiki. [Online]. http://microformats.org/wiki/rel-license

[66] (2012, March) W3C XML Query (XQuery). [Online]. http://www.w3.org/XML/Query/

[67] (2012, March) XML Query Language (XQL). [Online]. http://msdn.microsoft.com/en-us/library/aa924039.aspx

[68] (2012, March) Yahoo! Query Language - YDN. [Online]. http://developer.yahoo.com/yql/

[69] (2012, February) XML API reference - Google Custom Search APIs and Tools - Google Code. [Online]. http://code.google.com/intl/en/apis/customsearch/docs/xml_results.html

[70] (2012, March) SearchMonkey - Site Owner Overview - YDN. [Online]. http://developer.yahoo.com/searchmonkey/siteowner.html

[71] (2012, March) Google Webmaster Central. Get data about crawling, indexing and search traffic. Increase traffic to your site. [Online]. http://www.google.com/webmasters/

[72] (2012, March) Google Accounts. [Online]. https://accounts.google.com

[73] (2012, March) Providing Structured Data - Custom Search — Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_data#pagemaps

[74] (2012, March) HTML meta tag. [Online]. http://www.w3schools.com/tags/tag_meta.asp

[75] (2012, March) Providing Structured Data - Custom Search — Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_data#page_dates

[76] (2012, March) Providing Structured Data - Custom Search — Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_data

[77] (2012, March) Webmaster Tools - Rich Snippets Testing Tool. [Online]. http://www.google.com/webmasters/tools/richsnippets

[78] (2012, April) Filtering and sorting search results - Custom Search — Google Developers. [Online]. https://developers.google.com/custom-search/docs/structured_search

[79] (2012, March) Webmaster Tools - Rich Snippets Testing Tool. [Online]. http://www.google.com/webmasters/tools/richsnippets?url=http%3A%2F%2Fllrcp.ac.in%2Fmicroformats%2Ftest.html&view=cse

[80] (2012, April) Refinements - Custom Search Help. [Online]. http://support.google.com/customsearch/bin/answer.py?hl=en&answer=70359

[81] (2012, April) JSON. [Online]. http://www.json.org/

[82] (2012, March) Google Custom Search. [Online]. http://www.google.com/cse/compare

[83] (2012, April) Google Enterprise Search - relevant, easy search for intranets and

websites. [Online]. http://www.google.com/enterprise/search/products_gss_pricing.html

[84] Prince Bhanwra and Seema Bawa. (2012, April) Microformats based website - Thapar University, Patiala. [Online]. microformats.co.nr/Personalized User Profile Structure.pdf

[85] (2012, April) Yahoo! Search BOSS - YDN. [Online]. http://developer.yahoo.com/search/boss/

[86] (2012, March) HTML div tag. [Online]. http://www.w3schools.com/tags/tag_div.asp

[87] (2012, March) HTML span tag. [Online]. http://www.w3schools.com/tags/tag_span.asp

[88] (2012, March) Microformats. [Online]. http://microformats.org/

[89] (2012, March) hProduct · Microformats Wiki. [Online]. http://microformats.org/wiki/hproduct

List of Publications

I. Prince Bhanwra and Seema Bawa, "Microformats for adding Semantics on Web", communicated to "Internet Computing, IEEE", June 2012