

Talking Agent* : An Interactive Natural Language Query Interface to Raw Information

- Rohit Kumar -

10. August. 2003

Notes II: Web Crawler, Annotation and Information Warehousing Modules

As shown in figure 1 of Initial Notes, the basic blocks include Web Crawler and Information Warehousing modules. Also Text Corpus is required to be annotated manually and/or automatically. Discussed ahead is the proposal for developing these modules so as to be useable in the whole system considering the various requirements of the other blocks and the overall aim of the system.

We are now assuming that for all further notes we will work with Hindi Language encoded using UTF-8 standard and the data will be data from <http://www.bbc.co.uk/hindi/>.

Accessory Module: Fonts Converter and Related Issues

In accordance with these the first task would be to gain an understanding of UTF-8 encoding and its conversion to UNICODE. The character ordering in UNICODE is phonetic [Unicode 4.0 pg 222]. So its conversion to a format for use with Synthesizer will be quite simple. However, the complexity is then in rendering the font which we are not bothered about. Most platforms including Windows and Linux are providing full support to Unicode rendering even for Indian Languages. So a web based interface to this application is all that we would need to display the content too properly. Or we can write a converter from UNICODE to more easily displayable font like SHASHI or SHUSHA.

UTF-8 is basically a one to one notation of UNICODE. The conversion can be easily implemented using simple Boolean arithmetic. UNICODE is required to be stored as UTF-8 to avoid problems due to termination of string by presence of NULL (\0) character.

Brief Description of BBC Hindi Website and its Parsers

The BBC Hindi Website is very well built one with a lot of futuristic planning put into it. Also the data is largely consistent and correct. Basically the BBC sites are in 2 levels. The first level contains the headline of the most recent news of a particular area like General News, Sports, Science, etc. These headlines are accompanied by the following elements.

1. A link to a detailed article on that headline
2. A image file related to that article (OPTIONAL)
3. A brief description of the story of the headline
4. None or upto 3 Also See Links to articles relevant to the particular story related to the headline

At the first level there may be around 10 to 12 headlines per page. The second level is the descriptive pages of each of the headline. The link to this second level is got from first element as listed above. Also the Also See Links mentioned in fourth element above are links to pages in Second Level format. The pages in the second level consist of detailed story of the headline along with another or a couple of photographs.

Also each first level page consists of Date of their upload and other relevant information. Now the best part of BBC Hindi website is 2 things. First is that it support a standard encoding UNICODE in the UTF-8 representation. Second, the content on BBC Hindi Website is embedded with SPAN Tags. The SPAN Tags specify the nature of content it follows and also information like alternative text for images is provided appropriately with IMG tags. The presence of these tags makes the job of Annotation and Parsing the BBC Hindi web-pages much easier.

Second Accessory module required for the project is two Parsers along with a HTTP Client module which can download the content on the website given the URL of a first level page. The Parsers would parse the files using different rules depending upon the level.

Web Crawler

So the web crawler is basically a cron job which will be started once per day with a set of first level BBC Hindi Website URLs. After downloading each first level page, it will be parsed by the First level parser and only the relevant information i.e. class property of the SPAN tags and the associated content and also related Link URLs with proper tagging is stored. The image links are downloaded and stored in graphics folder (As will be described later) and the corresponding information like Alternative text, location of offline storage, article associated with is stored with the output of parsed content. Also this image information is stored in the Graphics object directory along with the date information.

The URL links found on the first level pages i.e. through the link at the headline and through the Also See links are stored in a master directory along with information about where the offline content corresponding to that particular content on that particular has been stored. The master directories for first level and second level are separate.

Then the second level pages linked from first level pages are downloaded by the HTTP Client module. But before this it is checked in Second level master directory whether those particular have already been parsed earlier. If so then no downloading takes place. Only the entry for offline location of the content is set appropriately. If the second level page is not yet downloaded, it is downloaded and then parsed by the second level parser. The information parsed by the second level parser includes the detailed story and related images. The images are stored as mentioned in the case of first level images. Similarly the storage of Also See links is done.

Also an accessory module to generate unique ID for offline storage of content is needed. The module should generate a sensible but still unique name for all image files and also. Also module for generating unique name of offline storage folder depending on the date of download should also be there.

Offline Content Directory Structure

The offline content will be stored in a directory structure as described ahead.

```
/
/ - - - Master Level 1 Directory (Dates with Folder Name containing that date's content)
/ - - - Master Level 2 Directory (URL with the locate of Offline storage)
/ - - - Graphics Directory (URL, Location of Offline Storage, Offline Location of Linked
                           from Page, Alt. Text)
/ - - - Graphics Folder (contains lot of JPG with unique file name listed in
                           Graphics Directory)
/ - - - Offline Web Content Directory
  - - - > / Date wise folder names
    / - - - Output of First Level Parser of the Main page of that date
    / - - - Detailed News Content folder
      - - - > Output of Second Level Parser with Unique File Names
      - - - > ...
      - - - > ...
  - - - > ...
  - - - > ...
```

Annotation of the Downloaded Content: Automatically and Manually

The content generated in the <Date Wise Folders> is to be annotated to mark the keywords. Now simple keyword detection (or may be even some complex Keyword detection algorithm) would separate the noise words from the Keywords. Later on only simplistic alternative left with us is that we index all the words identified by the keyword identification module and then search through them when required. The approach mentioned below (I believe) would be more suitable for finding most relevant text for Natural Language Queries and to answers queries like Where, What, Why, When more effectively.

We annotate the text collected offline in the Output of both first level and second level parser with tags that specify the type of content element that text enclosed in that tag is. The content element that we are interested in tagging are Name of Places (New Delhi, Beijing, Chandigarh, Juhu Beach, etc.), Name of Important People (Vaajpayee, Clinton, Rolling Stones), Name of Events (Election Campaign, Football Match, Rock Concert, Student Festival), Relevance of content (political, economic, sports, scientific, etc.), Mode of Transport (Airway, Submarine), Occupation (Scientist, Politician, Filmstar, Sportstar) and many many more of these content elements. These are infact very crucial elements and lot of analysis of the kind of question that we expect to find out

answers will decide what all these content elements are exactly going to be. Also a general content element may be defined so that significant words which do not fall into any particular content element category can be put into it.

Now the question comes is that how do we say that a particular element is belonging to a particular category. Here the Manual as well as Automated work comes in. Best and simplest approach is to build up dictionaries of content that may belong to each of the possible content element. And these dictionaries we will keep on updating daily as we find more and more words in the downloaded content that belong to a particular directory category. We may soon (in a couple of months) reach a good enough amount of information about these. This will involve one person to read the downloaded web content daily (nearly 100 web pages of 20 sentences each) and to identify the elements which are not automatically annotated from the already identified content. We can build a small tool which can display UTF-8 content and will allow selecting words/phrases and tagging them.

Further we may consider hierarchy of relations and those kinds of thing within the categories. Like Famous Personalities could be a super set of categories like Rock Star and Politician. Some WordNet based techniques etc. may be applied. And ofcourse some NLP based solution to the Annotation almost completely automatically can be applied. Finally if lot of interest is there then the content can be UNLized or may be tagged with some CYCs kind of things to build up natural language relations. But for the simplest approach we can easily build dictionary as already mentioned.

So the final offline content may be in some format like the following.

```
<DATE>
10 <MONTH> August </> 2003 <DAY> shaniwaar </>
<DATE>

<CLASS = HEADLINE>
<POLITICIAN, IMPORTANT PERSON, POET> vaajapayee </> aaj
<PLACE> Beijing </> paunche
<CLASS>

<CLASS = BRIEFSTORY>
<DESIGNATION> pradhaan mantra </> <IMPORTANT PERSON, POLITICIAN, POET> shri atal bihari
vaajpayee </> apne 3 din ke daure pe aaj <PLACE, COUNTRY> china </> ki raajdhani <PLACE, CITY> Beijing
</> paunche. Unhone <PLACE, COUNTRY> China </> ke <DESIGNATION> pradhan mantra </>
<IMPORTANT PERSON, POLITICIAN> shri ching shung fung </> se bhaint ki. Kal wo <PLACE, TOWN>
Wuhan </> me aayojit <EVENT> samelan </> dekhne jaayenge.
<CLASS>
```

Shown in figure 2 is a block diagram of the system elements involved in downloading the content from BBC Hindi website and storing it with the annotation in the offline content directory structure as described in this document. It is quite self explanatory. Essentially, it is the block diagram of Web Crawler with some additional components that are required for proper offline storage of news text corpus with annotations.

Data from BBC Hindi website is downloaded by HTTP Client and is then parsed by suitable Parser. Level 1 Parser may request further downloads. Level 2 parser can also

request Image downloads. The converters may or may not be used depending upon choice of storage notation. Further Directory Lists and folders are maintained as discussed.

B. B. C.

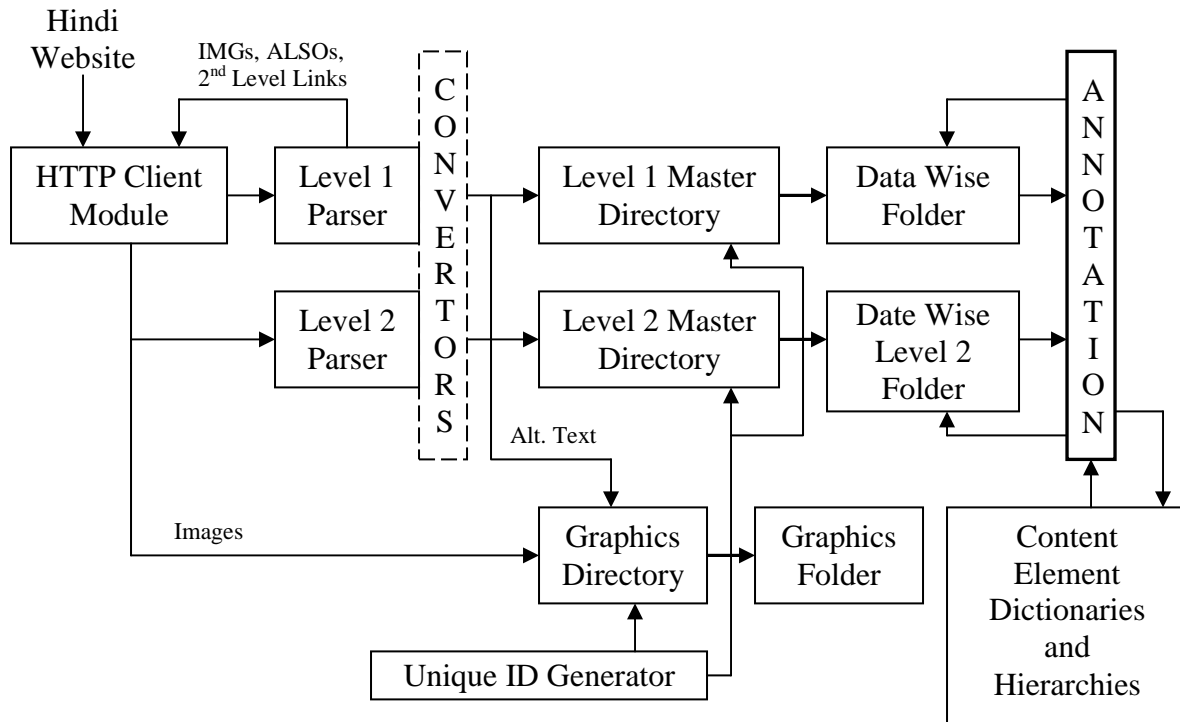


Figure 2: System of building up the offline news corpuses with Annotations

Indexing the Annotated offline data

Information Warehousing Module as mentioned in Figure 1 basically comprises of a module that generated indexes of the downloaded data. A possible indexing scheme that we may consider for using is described ahead.

All the keywords that get annotated after download (either manually or automatically) will be indexed. Here the significance of the generic content element category may be noted. We should maintain indexes in various dimensions. Basically there should be three indexing dimensions.

1. Date wise
2. Alphabetic (or may be Unicode value wise)
3. Content Element Category wise

For this purpose, we can assign a number each Content Element (see ahead) and a unique number is already assigned to each date using the Unique ID generator and alphabetic is already a number. Then the whole indexes are like a 3 dimensional space. If x-dimension is Date and y-dimension is Content Element Category and z-dimension is the Alphabetic order, then we can believe us to be in (-, +, +) octant. The date dimension will always count back and the axis will always be on the present date. So given a feature

vector of the form [date, category, keyword] (where each one of the 3 features may be specified in various possible ways like using wildcard or ranges etc.) we can localize the volume of our search as to find out the most optimal thing we are looking for.

Each available point in the 3 dimensional space will be a keyword along with its frequency count in the document from which it is taken and an address or identifier to the location where its offline data has been stored. Also we can allow default value in the 3 dimensions. Like if no date is given, we can search for the latest dates. If no category is given we must apply a module that will look for the given keyword (if any) to find its category or to recommend the generic category.

The more the detail about what is wanted is given the smaller the volume of our search space will be and that will help in find better results. The role of the NLQP module and to some extent the IR module will be to make these input parameters as specific as possible. I have some ideas for it and will discuss them when I write notes on NLQP and IR modules. One thing we can visualize is that once we are given the query parameters, the basic IR job will be use some optimization function based on the date and the frequency count and also the Boolean elements in the query to generate values for the points in the vicinity of search volume. Hence we will be plotting a kind of contour or 4 dimensional plot where the fourth dimension will be the value calculated by the optimization function. We can see it to quite an extent like unit selection. The solution would be to a peak finding problem.

In between: To assign number for each content element category we need to consider also the cases where a particular keyword would belong to multiple categories. One simple thing that we can do is to create a list of all possible combinations in which a word may have category combinations (this will be limited because a mode of transport element cannot be a politician element). Then we can assign unique ID to it all. Alternative a more intuitive scheme may be designed based one bit per category type etc. like the one used to classify phones in my IED paper. Many more simple technique can we thought of.

Now the issue lies in how to store the data to represent a three dimensional space and allow easy search and localization depending on a criteria that can we specified in a large variety of ways. The simple answer would be to use a database which allows a great interface to query it. May be other complex alternatives exist but there a lot of effort would be involved in developing already well established data structures and algorithms.

The indexing i.e. warehousing module should be so designed that depending on some parameters it should be able to index the whole data available offline or a certain part appending to the databases. It will be required because if we allow only indexing to new data then when adding new content element categories or when changing the indexing, we will leave out the old data which we don't want.

There will be several implementation issues when we get to implementing it but those will be handled if the basic guidelines in these notes are tuned further. In the next notes I ll discuss the ideas about NLQP and IR modules. There are several issues with them.