Abstract - To develop the Database module of Vijjana and visualize the data in the form of some standard Views. According to the standards of IEEE 1484.12.1, a main and well structured relational database of Vijjana is built using MySQL. The database is populated by transferring data from an Open Directory Project (ODP). Data is extracted out of it in the form of an XML and is viewed in the form of a Tree View and a Radial View using visualization techniques.

Keywords: Vijjana, Database, Jan Structure, ODP, Prefuse.

1 Introduction

Web plays a vital role in our daily life, a person x will surf through the web daily when he needs information about books, tutorial, health, news, sports shopping and entertainment. Etc. Information on a particular topic is scattered all over the web at various sources each millions of pages away from one another, to cope up with this people use search engines to get all the information on the topic displayed at one place.

There are many search engines like Google, Yahoo, MSN, AOL, EBAY, Netscape etc. Some search engines relies on searching data over the web using keywords specified for specific web pages, but search engines like Google will search the pages through the description of it rather than the specified keywords. This is the reason Google is well used by many users. Even though these search engines are able to provide large amounts of pages to the user, the user still has to surf through the large number of results to find the most appropriate result for his problem. The same person will end up in finding the result sooner if he has a general idea about each and every result or the result which worked the best for his friends working on the same topic etc. On the whole this problem can be solved by developing an agent which could successfully handle the results bound to particular topics or domains and which can also provide the personal opinions of users who have worked or are working on the same topics by saving their comments or rating and enabling more collaboration between them. This framework called Vijjana[2] is going to break through all these problems by providing

Pragmatic mechanism for collaboratively building useful knowledge networks in well-bounded domains.

2 Problem Statement

This paper illustrates the way we develop the Database module of Vijjana and visualize the data in the form of some standard Views. According to the standards of IEEE 1484.12.1, a main and well structured relational database of Vijjana is built using MySQL. The database is populated by transferring data from an Open Directory Project (ODP) [3]. Data is extracted out of it in the form of an XML and is viewed in the form of a Tree View and a Radial View using visualization techniques.

3 The Vijjana Model

We define the Vijjana model as:

\[ Vijjana-X = \{ J, T, R, dA, oA, cA, vA, sA, rA \} \]

where

\[ X = \text{the domain name} \]
\[ J = \text{the collection of Jans in the Vijjana-X} \]
\[ T = \text{the Taxonomy used for classification of Jans} \]
\[ R = \text{the domain specific relations} \]
\[ dA = \text{the discovery agent which find relevant Jans} \]
\[ oA = \text{the organizing agent which interlinks the Jans based on R} \]
\[ cA = \text{the consistency/completeness agent} \]
\[ vA = \text{the visualization agent} \]
\[ sA = \text{the search agent} \]
\[ rA = \text{the rating agent} \]

The markup agent is a sub-agent of the discovery agent. Similarly, the validation agent is a sub-agent of the consistency/completeness agent. We now examine the underlying concepts followed by the markup process and the validation process.
4 VIJJANA DATABASE IEEE 1484.12.1

This is a multipart standard that specifies learning object metadata. This part specifies a conceptual data schema that defines the structure of a metadata instance for a learning object. For this standard, a learning object is defined as any entity—digital or non-digital—that may be used for learning, education, or training. For this standard, a metadata instance for a learning object describes relevant characteristics of the learning object to which it applies. Such characteristics may be grouped in general, life cycle, meta-metadata, educational, technical, educational, rights, relation, annotation, and classification categories.

Metadata is information about an object, be it physical or digital. As the number of objects grows exponentially and our needs for learning expand equally dramatically, the lack of information or metadata about objects places a critical and fundamental constrain on our ability to discover, manage, and use objects. This standard addresses this problem by defining a structure for interoperable descriptions of learning objects. A data element for which the name, explanation, size, ordering, value space, and data type are defined in this standard is known as a Learning Object Metadata (LOM) [7].

Data elements describe a learning object and are grouped into categories. The LOMv1.0 base schema:

The general
Category groups the general information that describes the learning object as a whole.

The lifecycle
Category groups the features related to the history and current state of this learning Object and those who have affected this learning object during its evolution.

The meta-metadata
Category groups information about the metadata instance itself (rather than the learning object that the metadata instance describes).

The technical
Category groups the technical requirements and technical characteristics of the learning object.

The educational
Category groups the educational and pedagogic characteristics of the learning object.

The rights
Category groups the intellectual property rights and conditions of use for the learning object.

The relation
Category groups features the relationship between the learning object another related learning object.

The annotation
Category provides comments on the educational use of the learning object and provides information on when and by whom the comments were created.

The classification
Category describes this learning object in relation to a particular classification system.

5 data layer

5.1 Source Data for Database

As we discussed earlier, information or knowledge is scattered all over the web at different places. As users search on a particular domain or topic we need to have a collection of URL’s for each and every domain. So a resource with a predefined taxonomy with collection of Jan’s for each and every particular domain is necessary. Here Open Directory project (ODP) on Dmoz.org, the largest and most comprehensive Human-Edited Directory of the web with well structured and predefined taxonomy, is used as the core dataset for the database of Vijjana. The Open Directory follows in the footsteps of some of the most important editor/contributor projects of the 20th century. Just as the Oxford English Dictionary became the definitive word on words through the efforts of volunteers, the Open Directory follows in its footsteps to become the definitive catalog of the Web. The Open Directory was founded in the spirit of the Open Source movement, and is the only major directory that is 100% free. There is not, nor will there ever be, a cost to submit a site to the directory, and/or to use the directory's data. The Open Directory data is made available for free to anyone who agrees to comply with our free use license.

The Open Directory [3] powers the core directory services for the Web's largest and most popular search engines and portals, including Netscape Search, AOL Search, Google, Lycos, HotBot, Direct Hit, and hundreds of others. The source of Data is directly form the user himself, when as we discussed whenever a user Mark-Up's a Jan then that Jan is automatically added to the database if it Is not present in the database and will be added in his Jan's list. Also the Consistency Agent will periodically check the Jan's present in the Database and removes dead links so the database shrinks and grows from time to time.

The database of Vijjana consists of:

1. Information or properties of the JAN’s and the
2. Information of the user and the
3. Relation of Each Jan with user like
   a) Who added it
   b) How many times it has been marked up by users.
   c) When it has been last modified
   d) Rating
It also shows the Present Status of JAN whether it is an alive or a dead link.

5.2 JAN Structure

The Above picture shows the typical schema or structure of the Vijjana Database.

1. The tables General, Identifier and Classification are related to the Collection of the JAN’s form the Open Directory Project.
2. The Properties of the JAN are given by the tables:
   a) Educational
   b) Technical
   c) Rights
3. The Status of the Jan is given by the Table Life Cycle.
4. The User Information is given in the table Membership
5. The relation between the user and JAN is given by the tables:
   a) Comment
   b) CommentRating
   c) JanRating
   d) Keyword

5.2.1 Educational
This table describes the followings:

1. The educational and pedagogic characteristics of the learning object. The pedagogical information is useful for users who are involved in achieving a quality learning experience. The users for this metadata include teachers, managers, authors, and learners.

2. The degree of interactivity to which this learning object is categorized. Interactivity in this context refers to the degree to which the learner can influence the aspect or behavior of the learning object. The Interactivity type can be “Active” Learning, “Expositive” Learning or “Mixed” learning “Active” learning (e.g., learning by doing) is supported by content that directly induces productive action by the learner. “Expositive” learning (e.g., passive learning) occurs when the learner's job mainly consists of absorbing the content exposed to him when a learning object blends the active and expositive interactivity types, then its interactivity type is “mixed.”

5.2.2 Rights
This table describes the followings:

1. The intellectual property rights and conditions of use for this learning object.
2. Whether copyright or other restrictions apply to the use of this learning object.

5.2.3 Technical:
This table describes

1. The technical requirements and characteristics of this learning object. Technical data type of this learning object.
2. To identify the software needed to access the learning object. Information about other software and hardware requirements.

5.2.4 Lifecycle
This table describes the completion status or condition of this learning object.

5.2.5 Membership:
This table completely relates to the personal information of the user. It contains the registration and login details of every user. Apart form the field of password, the Security Question and Answer will enable more secure login. It can also be used when user forgets his password.

5.2.6 Comment:
This table stores all the comments given by the user about respective JAN’s. The field UserID shows the particular user, the field CommentID says the related comment of that user and the field GeneraID shows the particular JAN. It also shows the date the comment is created and last modified.
5.2.7 Comment Rating:
This table shows the user ratings for each and every comment given by different users. For a particular comment given by a User, many other users can rate it according to their wish. The UserID shows the user who has rated it and the CommentID shows the comment on which he has rated it.

5.2.8 JanRating:
This table shows how much the users have rated for each and particular Jan. The UserID gives the information of the user who rated that JAN and the GeneralID gives the information of that particular JAN. Generally a user can rate any Jan irrespective of he has created it or not.

5.3 Mysql DATABASE and Phpmyadmin
All above are implemented using MYSQL database which is an open source database tool. We use its workbench tool to draw the ORM model and another open source tool, phpmyadmin, as our web interface to monitor the database.

5.4 Populating the Database
As we have already chosen to use the taxonomy of the Open Directory Project (Dmoz.org) as the core data for the Vijjana Database, the task now is to transfer the data from ODP to our Database [8]. The data in the ODP is in the format of RDF [4] known as Resource Description Framework. Files in this form will use tags to distinguish different resource categories. To transfer these files into a database, the editor of Open Directory Project created a set of tool named “ODP/dmoz”. A famous tool for this is called “PhpODPWorld”.[5].

The script used to import the RDF file into database is developed by Steve who is one of open directory project editor. The following procedure explains in steps to do the RDF to Database transformation:

1. To the wanted directory on the web server, extract and move all the unpacked files of downloaded PhpODPWorld package.
2. Complete the following steps
3. Create "logs" directory, if you enabled logging in the config file and this directory must be writable by the web server
4. Create "smarty/cache" directory and "smarty/compiled", if you enabled smarty in the config file and this directory must be writable by the web serve
5. Create a database and a user either with password or without.
6. The database table defined in "tools/db.sql" is created.
7. Wanted categories and references, database settings are reflected by editing "config.inc.php" and "tools/config.pl".
8. Downloaded either the complete content or structure RDF.
9. Use the Perl script "tools/extract.pl" to extract your categories from the RDFs. (Perl module DMOZ- ParseRDF-0.14 should be installed now)
10. Using Perl scripts insert your categories (from the RDFs) into the database.
11. Do the following for initial run to update the count (of sites) for each category
   a) Turn on maintenance mode in "config.inc.php".
   b) Turn off maintenance mode in "config.inc.php".

5.5 Data Structure
After populating the database the data has been moved into three main tables known as:
   • Classification
   • Identifier
   • General

5.5.1 General

<table>
<thead>
<tr>
<th>GeneralID: INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title: VARCHAR(255)</td>
</tr>
<tr>
<td>Language: VARCHAR(45)</td>
</tr>
<tr>
<td>Description: VARCHAR(255)</td>
</tr>
<tr>
<td>Keyword: VARCHAR(255)</td>
</tr>
<tr>
<td>Coverage: VARCHAR(255)</td>
</tr>
<tr>
<td>Structure: INTEGER</td>
</tr>
<tr>
<td>AggregationLevel: INTEGER</td>
</tr>
<tr>
<td>OwnerID: INTEGER</td>
</tr>
<tr>
<td>MarkupCount: INTEGER</td>
</tr>
<tr>
<td>Valid: INT</td>
</tr>
</tbody>
</table>

Figure 1: General Table Structure

The most important fields of this table are General ID, Title, Coverage and Aggregation Level. General ID describes or gives the General ID for each and every node in the tree or graph and at the end of the taxonomy it gives the ID for the leaf. Title Gives the Name of the node of the graph and at the last level it gives the name of the leaf. Coverage is the most Important field of the table and it shows the taxonomy through which this node or particular Level is achieved. Aggregation Level shows the number of Childs a particular node (Root in this level) has.
Owner ID gives the ID of the person who has added this JAN. Also the no of times that particular JAN has been marked up is given by Markup Count. Valid field shows the status of the JAN whether it is alive or a dead link. The keyword field stores the keyword given by the USER for his own convenience.

5.5.2 Classification

The Classification ID gives the ID into which that particular NODE has been classified. GeneralID refers to the general Id of this node or leaf. User Id gives the ID of the user who classified the JAN. KEYWORD stores the keywords given by that particular User.

5.5.3 Identifier

This is the table where the Value of the JAN or the URL will be stored. Identifier ID is a special ID here which is given for the END leaf or the URL of the whole taxonomy. This ID is different from the GeneralID and will not have any values for the nodes. The foreign Key GeneralID gives the general id for this Node (Leaf in this case, since the last level)

Note: Identifier ID gives an ID for the end URL’s or Jan’s (Leafs) only in the Taxonomy (Tree) and GeneralID gives the ID for both Leaf’s and Nodes.

Entry gives the value of the URL or JAN. Title here gives the Title for the JAN and description gives about the information of the JAN. Invalid Count gives the no of times this JAN has been Invalid or Dead. Status gives the present condition of the JAN if it is ALIVE or DEAD. LastModifiedField gives the last time this JAN has been modified.

6 Visualization Data

6.1 Introduction

Wiki’s Definition of Visualization is any technique for creating images, diagrams, or animations to communicate a message. Visualization through visual imagery has been an effective way to communicate both abstract and concrete ideas since the dawn of man. Visualization [6] of Vijjana is the most important part of all the agents that we have in Vijjana model. It is the users interface to interact with the knowledge base to find the semantics [1] and also to obtain relevant information in a particular field by means of user friendly navigation.

Now for the Visualization of Vijjana we have to first connect to the database of Vijjana and generate graph XML files, which are given as an input to the Prefuse and Hypergraph tools and thereby we will get corresponding User Controllable visuals. Before dealing with generation of XML, I will introduce XML, XML schemas and the XML schema’s used for different visual views.

The raw data from the database is to be given in an abstract form to the Prefuse & Hypergraph toolkits, so an XML having the structured information about the Vijjana semantic net or taxonomy is required. There are several XML schemas that are related to the corresponding visual views. For example for Hypergraph we use an XML that has the schema of GraphML. So based on the type of Visualization that we generate, first we have to study the XML schema and then we have to transform the raw data to that form of XML.

The XML file is easy to transfer because of its small size but, hard to interpret because of its flexibility. Actually, what we are going to implement is to create the XML data schema based on the database table we have, and create an automatically intelligence system to generate the XML file based on the search result. The result is large XML file which can be interpret based on several of data schema, and it will return different small xml files based on them. This XML file can be interpreted as the raw data file for the visualization and other purposes.
Now for the visualization of Vijjana, in the first stage, we have to define the logic in retrieving data to follow the rules of XML schema’s of corresponding visualizations.

6.2 MySQL Query Browser

The MySQL Query Browser is a graphical shell where you can execute queries and develop SQL scripts, with several features to help you improve your productivity. The MySQL Query Browser interface tries to mimic the interface of a web browser. [9]

Here we are using MySQL query browser to show the way we query the data from database. Queries are executed on this local database to extract data in the desired manner for the generation of the XML’s. These XML’s are used as an input for the Prefuse to generate the Visualization techniques. Since the MySQL query browser interface almost acts as the original browser, our task is to fill the local database with the data present in the original database over the server. After the data is ready in this database, several operations can be done to manage it and it also can be accessed in different programming languages for generating the XML’s.

The whole process in creating a Database and populating it with data is described below:
1. Install MySQL Server 5.0
2. Open MySQL query browser
3. On the top of the page you can see a small window (Query Execution Space), execute a query for creating a new database.

```
Create Database Vijjanadatabase;
```

**Figure 4: Query Execution Window of MySQL Query Browser[9]**

4. The created database will be shown at the schemata window. Select the database by double clicking it; this means that all the future operations like queries etc will be done on that database since you have selected it.

```
```

**Figure 5: Database Is Selected**

5. Now create a new Table by name general in the database (Vijjanadatabase) by executing the whole SQL script for the Table General at the query window.

6. The created table will not be shown immediately, so right click on the database Vijjanadatabase and then click refresh. This can also be done by double clicking on the Vijjanadatabase.

7. Likewise all other corresponding Tables which are present in vijjanadatabase are created using their corresponding scripts.

```
```

**Figure 6: Showing the Contents present in Vijjanadatabase**

8. Now select the general table by double clicking it. This also generates the query automatically in the query browser window. At the start the General table will not show any values because it not having any data with it. This is similar with all other tables so created.

```
```

**Figure 7: Query Automatically Generated When General Table Is Double-clicked**

9. Now our task is to populate this local database with the data present in the server. To do this we have to first download all the data from the server from eksarva.csee.wvu.edu/vijjanadata.

10. The above site would let you download over 1GB of data file with name “vijjanadata”, Change the name of this file to “vijjanadata.sql” so that it will become an executable SQL file format.
11. To import this data into the database on your machine (vijjanadatabase), we use cmd prompt. “mysqlclient -u root -p vijjanadatabase < vijjanadata.sql.”

12. The above command will dump the data vijjanadata.sql in the vijjanadatabase created on your local machine. A dialog box will appear showing Dump is complete. This means that the data has been imported into the local database and is available for usage.

After the data has been imported into the local database (Vijjanadatabase), it can be used for further execution of queries and for generating the XML’s required for the visualization[6].

Now programming and query execution can be done on this local database rather than working on the large some of data over the server.

8 References


