International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:5, No:12, 2011

Mining and Visual Management of XML-Based Image Collections

Khalil Shihab and Nida Al-Chalabi

***Abstract*—**This article describes Uruk, the virtual museum of Iraq that we developed for visual exploration and retrieval of image collections. The system largely exploits the loosely-structured hierarchy of XML documents that provides a useful representation method to store semi-structured or unstructured data, which does not easily fit into existing database. The system offers users the capability to mine and manage the XML-based image collections through a web-based Graphical User Interface (GUI). Typically, at an interactive session with the system, the user can browse a visual structural summary of the XML database in order to select interesting elements. Using this intermediate result, queries combining structure and textual references can be composed and presented to the system. After query evaluation, the full set of answers is presented in a visual and structured way.

***Keywords*—**Data-centric XML, graphical user interfaces, information retrieval, case-based reasoning, fuzzy sets

I. Introduction

AS two tendencies for the current and future development of image libraries are crucial for the purpose of this paper. First, the amount of images and text used for the description of these images in a typical digital images-based database grows continuously. For example, Image Database at the College of Veterinary Medicine, Washington has a large collection of Veterinary and animal related images [1]. The Defense Image Database, which is an official Ministry of Defense, UK, holds thousands of images [2]. The virtual museum of Japanese art is a comprehensive site includes information about and photos of all forms of traditional Japanese art [3]. Fine Arts covers painting, sculpture, and ukiyoe; Crafts introduces ceramics, textiles, and metalwork; Performance Arts introduces Kabuki, Noh, Bunraku, and Kyogen; Pastime Arts explores Bonsai, Ikebana, calligraphy, and tea ceremony; and Martial Arts provides an overview of Sumo, Judo, Aikido, Kyudo, etc. The Web Gallery of Art is a virtual museum and searchable database of European painting and sculpture of the Romanesque, Gothic, Renaissance, Baroque, Neoclassicism, and Romanticism periods (1000-1850), currently containing over 27.000 reproductions [4]. The art history virtual library contains more than 40000 Images [5].

Khalil Shihab is with the University of the South Pacific, School of Computing, Information and Mathematical Sciences, Suva, Fiji (phone: +679-3232329; fax: +679 323 1527; e-mail: khalil.shihab@gmail.com).

Nida Al-Chalabi, is with Sultan Qaboos University, Department of Computer Science, Muscat, Oman (phone: +968 2414 2400; fax: +968 2414 1407; e-mail: nida@squ.edu.om).

The tourism image Australia gallery depicts the Australian people, environment and the lifestyle across a range of experiences including nature, indigenous culture, outback, adventure, beach, cities, and food and wine [6].

In some of our own experiments, we used a large collection of images from Iraqi Museum and from the Oriental Institute of The University of Chicago [7]. We used XML documents to annotate and store 160 images up to 10.3 MB, ranging from prehistoric period to the Achaemedian and Seleucid 500 BC. These artifacts include many of the most famous works of ancient Sumerian, Akkadian, Babylonian and Assyrian art. In particular, they include the Uruk vase, dating from 3500BC and artifacts excavated from the ancient Sumerian city of Ur.

XML documents provide users with a mean to store and deal with valuable information on a wide range of domains. XML and databases are a natural fit for each other in three important ways. First, XML documents provide a platform-neutral mechanism for transporting data between databases and applications. Second, databases provide an efficient way to store and query XML documents. The third way is the loosely-structured hierarchy of XML documents provides a useful representation method to store semi-structured or unstructured data, which does not easily fit into existing database models. These characteristics encourage researchers and companies to develop many XML-based databases that allow preserving physical document structure, support document-level transactions, and execute queries in an XML query language. However, the increasing use of a large number of XML documents causes many problems to the users [8]. In particular, the structure of these XML documents adds an additional problem in dealing with them. One of the problems of XML documents is the searching that can be too complex for most users. XML documents are generally not interoperable in the same search environment, because of all the different, incompatible vocabularies. XML searching requires people or software to know a lot about the structure of the documents. Moreover XML does not have any browser support and does not have anything to support the end user applications. Therefore, automatic graph drawing is a necessary solution of these problems. It has many important applications in software engineering, database and web design, networking, and in visual interfaces for many other domains.

Although XML is good for data exchange between applications, it is often not chosen for visualization of the data because it is not very human readable. The second challenge is, therefore, to allow users dealing visually with XML documents. In this work, we describe the visual and interactive exploration of XML documents. We focused on the users and developed more intuitive ways to visualize the XML

documents. In order to visualize the informative content of an XML document, the structure of data has to be carefully preserved. Therefore, we used a tree-like structure in which nodes are used to represent the children (nodes) of the XML document and links between these nodes are used to represent the relations between these nodes [9].

The system consists of two main components: the Text Interaction Component (TIC) and the Visual Interaction Component (VIC). The users can use either component for creating an XML document, delete or add a node. In a typical session with Uruk, the user first creates a new or imports an existing XML database. After that, the user can browse a visual structural summary of the XML database in order to select interesting elements.

II. Visualization Features

The main features of the Visual Interaction Component (VIC) are as follows:

A. Simple Navigation with Minimum Disorientation

The graphic manager part of VIC was designed to include features to aid the user in navigating the visualization. In particular, if the XML document contains a large number of nodes, the graphic manager displays only the root of the tree. The navigation of any part (level) or the whole tree is left to the user. Therefore, the graphic manager provides the user a full control on the way he/she likes to be displayed and to work on.

B. High Information Content

The graphic manager allows the user to display the content of any node of the tree by moving and clicking the mouse on that node.

C. Low Visualization Complexity, Well Structured

If an XML document has a complex structure, the graphic manager displays not only the top level of the tree but also it displays the parent nodes of that level. The manager allows the user to explode these nodes to their children nodes, i.e. using partial display of the tree; the system provides the desired information to the user.

D. Resilience to Change

The graphic manager allows changes of content of any node and provides an option to the user for saving or ignoring the changes. In case of updating and then saving the resulting tree, the system maintains the integrity of the data structure of XML document.

E. Good Use of Interaction

The system provides a pull down menu of a few top level options. Each of these options contains a few low level options. Therefore, the system is designed and implemented to be user friendly and easy to use.

It provided users with the following key features:

1) *Uruk is platform independent; it is developed in Java and can be used on any platform (Windows, Linux, Mac, etc.) out of the box. Also, it can be used as a web applet to be integrated into web pages. Uruk occupies less than 50MB of the system memory when running.*
2) *Due to the shape of the nodes, Hydra and other existing visualization systems would not produce readable results when drawing large XML documents. Uruk uses specific algorithms which sort the nodes to be presentable to the human user.*
3) *Uruk allows the user generating an XML document visually, without any XML knowledge. It is technically referred to as "XML WYSWYG Editor".*
4) *Uruk is able to export the graph as Image and GraphML (XML-based file format for graphs) files. GraphML is a de facto standard for graph representation and this feature enables Uruk to collaborate with external graph drawing libraries such as yFiles, which is known as the world's best graph drawing library. Users are not bond to Uruk's graphical features when it comes to XML visualization; they could convert their XML files to GraphML by Uruk and then draw the GraphML file in their desired application.*
5) *Uruk is a multi-graph application. Therefore, users can open and visualize multiple XML documents simultaneously and work on them individually.*
6) *Uruk draws the graphs in multiple layouts (Tree and Circle are currently implemented; many more layouts are possible to apply).*

F. Mining XML Documents

There are two main phases in the development of this important part of the system. The first is the search and growth phase. Here, the ranking system first constructs a collection of nodes about a query string. Since the search results may contain a large number of nodes, this number must be limited to a reasonable quantity so that the system can reach a compromise between obtaining a collection of nodes highly relevant and saving computational effort. For constructing such a collection of nodes, the ranking system makes use of the results given by a text-based search engine. The search engine will return a set of nodes which are determined by its own scoring function as a root set. It then extends the root set by adding any additional nodes that is pointed to by a node already in the root set. The new collection is then renamed the base set. In this way, the link structure analysis can be restricted to this base set, which is expected to be relatively small, rich in relevant nodes.

The second is the weight and propagation phase, in which the results returned by the first stage are evaluated. Here, the ranking system calculates the rank score of each node based on the link structure between any node pairs in the base set, and extracts good authorities and hubs from the overall collection of nodes.

G. Computing similarity between nodes

The computing is divided into three steps, which are: Generating extended-element vectors, Measure of element similarity, and Constructing of the similarity matrix.

Generating the extended-element vectors for an XML document is as follows:

1) Parse an XML document to extract elements and generate a DOM (Document Object Model) tree.
2) Sift meaningful tokens by ?ltering delimiters such as space, hyphen, and under score.
3) Delete tokens included in a stop-list.
4) Extract stems or original form of the tokens through stemming process.
5) Extend elements thus found, using the WordNet thesaurus and a User-defined word library, with synonyms, compound words, and abbreviations.

III. The Reasoner Module of Uruk

An integrated approach that is based on case-based reasoning and fuzzy sets is used as the underline technique of the reasoned module. It makes use of past experiences to derive the solution for a new problem. It has been widely implemented in practical [10]. To process past experiences (cases) efficiently, a common case-based reasoning technique is to select some characteristics that are representative of the cases and use them as indexes to store the cases. Later, to solve new problems, the system uses these characteristics as probe to retrieve the set of similar cases that are then adapted and modified to arrive at a targeted solution. Often, it is a common practice to narrow the set of retrieved cases by means of a similarity metric. Another problem encountered in case-based reasoning is the acquisition of past experiences when the reasoner is initially deployed. At that early stage, the reasoner may have to find a solution from scratch due to insufficient numbers of past cases to be used as model. Therefore, we used XML as case representation for making up structured knowledge-rich data.

Using fuzzy indexing and retrieval allows attributes that are characterized by numerical values to be converted into fuzzy sets to simplify comparison. For example, the height of the artifact can be converted into categorical scale (e.g. tall/large, medium and short/small). Also, fuzzy sets allow multiple indexing of a case on a single value with different degrees of membership. For example, if the size is 60cm, this can be classified as tall with 0.4 and medium with 0.7, where 0.4 and 0.7 are the degrees that the height is classified as tall or medium respectively. This treatment increases the flexibility of case matching by allowing the case to be considered as a candidate when we are looking for an artifact with either large or medium size.

A. Fuzzy sets and membership functions used in the system

In fuzzy sets an object may partially belong to a set, so the set must be represented by a continuous membership function maps the domain of the set to an interval of [0, 1]. For example, the following functions (1-3) and Fig. 1 show the membership functions of high, moderate and low utilization as they are applied to the size and estimated price of an artifact in our application. Classical sets, which are subsets of fuzzy sets, represented by binary membership functions and therefore, they are subsets of fuzzy sets, [11, 12].

Since fuzzy sets use possibilities rather than binary membership values, a threshold value is often used to differentiate those considered highly likely to be a member of a set from those considered relatively unlikely. For example, when we are seeking for artifacts that have large size or tall, we may want to consider only those with membership grades of tall are above 5. This value is generally called α-cut. For example, if the membership function of tall, as defined in Fig. 1, is given and if the α-cut is set at 5 for tall, then artifacts with height greater than 55cm are considered tall, whereas artifacts that have their heights greater than 61 are considered very tall.

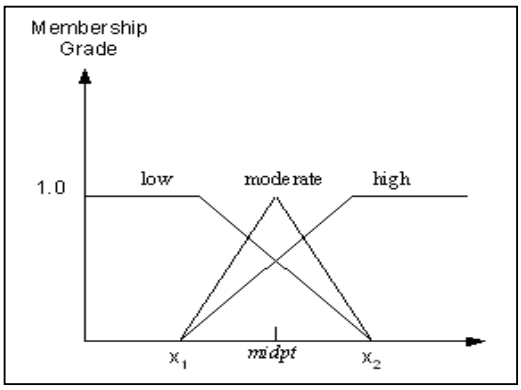

Fig. 1 The membership function of high/tall, moderate/medium and low/small

1. $\mu high(x) = (x-x_1)/d_o$ if $x_1 \le x \le x_2$, 0 if $x < x_1$, and 1 if $x > x_2$

2. $\mu moderate(x) = (x-x_1)/0.5d_o$ if $x_1 \le x \le$ midpoint,

$\mu moderate(x) = (x_2-x)/0.5d_o$ if midpoint $\le x \le x_2$,

$\mu moderate(x) = 0$ otherwise

3. $\mu low(x) = (x_2-x)/d_o$ if $x1 \le x \le x_2$, 0 if $x > x_2$, and 1 if $x < x_1$

Where x_1, x_2, d_o and midpoint are as follows:

$$x_1 = \begin{cases} x_{CPU} = 30 \\ x_{I/O} = 20 \end{cases} \quad x_2 = \begin{cases} x_{CPU} = 70 \\ x_{I/O} = 40 \end{cases} \quad d_o = \begin{cases} d_{CPU} = 40 \\ d_{I/O} = 20 \end{cases}$$

$$midpt = \begin{cases} midpt_{CPU} = 50 \\ midpt_{I/O} = 30 \end{cases}$$

B. Image indexing and retrieval

Case attributes can be either quantitative or qualitative. Qualitative attributes accept nominal values. For example, the artifact type is a qualitative attribute whose value may be stone, bronze/copper, clay, gold, ivory, or shell. Quantitative attributes, on the other hand, allow values to be measured on a numerical scale.

Fuzzy indexing and retrieval are useful in domains where cases have quantitative attributes. For cases with qualitative attributes only, indexing can be performed on attributes

directly. For example, artifacts can be classified as large, medium, or small (three classes according to their size); or can be classified according to their materials into six classes: stone, bronze/copper, clay, gold, ivory, or shell. We can easily index systems by their materials. If we also want to include the height or size, indexing becomes more complicated since the value of this attribute can be any positive real number. However, with a proper transformation into a few discrete classes based on practical requirements, indexing becomes easier to handle.

The process of fuzzy indexing is, therefore, of two stages. Quantitative attributes are first processed by the fuzzifier (called fuzzification) and then indexed on the resulting classes (indexing) before being stored in the CB. The following section describes these stages in more detail and illustrates how they can be applied to the lost treasures domain.

The fuzzification process includes the following steps:

1) When a case is encountered, qualitative attributes are identified.
2) For each quantitative attribute, proper classes are determined based on practical needs.
3) The membership function of each class and its associated α cut are determined.
4) Numerical values of each case are converted into proper classes for indexing.

To illustrate this process, a running example is used. The context is a lost treasure domain that contains Artifacts, Figurines, Inlays, Jewelry, Metal Vessels, Musical Instruments, Pottery, Relief, Seals, Sculpture, Vessels and Terracotta. They are categorized into six different types: stone, bronze/copper, clay, gold, ivory, or shell. Fig. 2 shows an example of XML document of some of these objects.

```
<?xml version="1.0" ?>
- <IMAGES>
  - <IMAGE>
      <SERNO>1</SERNO>
      <MuesumNumber>IM19755</MuesumNumber>
      <CATEGORY>Limestone, Female</CATEGORY>
      <MATERIAL>Limestone</MATERIAL>
      <KEYWORDS>Female, Standing,
      Limestone</KEYWORDS>
      <DESCRIPTION>Standing Female, Eyeballs of
      Shell</DESCRIPTION>
      <DIMENSION>HEIGHT/LENGTH/54cm, tall/0.62,
      medium/0.25, small/0.0.13 </DIMENSION>
      <LOCATION>Tell Asmar</LOCATION>
      <PERIOD>Sumerian, Early Dynastic II 2600
      B.C.</PERIOD>
      <STATUS>Stolen</STATUS>
      <URL>http://MySite/ImageGallery/Images/st
      anding_pic1.jpg</URL>
            </IMAGE>
  - <IMAGE>
      <SERNO>2</SERNO>
      <MuesumNumber>IM19653</MuesumNumber>
      <CATEGORY>Female, Standing,
       Stone</CATEGORY>
      <MATERIAL>Stone</MATERIAL>
      <KEYWORDS>Female, Standing, Stone, South-
       Iraq</KEYWORDS>
      <DESCRIPTION>Statue of female wearing
       elaborate, flounced garment
      leaving one shoulder bare </DESCRIPTION>
      <DIMENSION>HEIGHT/LENGTH/36cm, tall/0.54,
       medium/0.6, small/0.7    </DIMENSION>
      <LOCATION>Khafaji</LOCATION>
      <PERIOD>Sumerian, Early Dynastic II 2800
       B.C.</PERIOD>
      <STATUS>Unknown</STATUS>
      <URL>http://MySite/ImageGallery/Images/st
       anding_pic2.jpg</URL>
      </IMAGE>
  - <IMAGE>
      <SERNO>3</SERNO>
      <MuesumNumber>IM19759</MuesumNumber>
      <CATEGORY>Male, Stone,
      Standing</CATEGORY>
      <MATERIAL>Stone, Limestone</MATERIAL>
      <KEYWORDS>Male, Standing, Stone, South-
      Iraq</KEYWORDS>
      <DESCRIPTION>Statue of male bearded, long
      hair, bare-chested wearing
      flounced skirt, hands folded, standing on
      flat base </DESCRIPTION>
      <DIMENSION>HEIGHT/LENGTH/ 54cm, tall/0.57
      medium/0.46, small/0.42 </DIMENSION>
      <LOCATION>Tell Asmar</LOCATION>
      <PERIOD>Sumerian, Early Dynastic, 2600
      B.C.</PERIOD>
      <STATUS>Stolen</STATUS>
      <URL>http://MySite/ImageGallery/Images/st
      anding_pic3.jpg</URL>
  </IMAGE>
  - <IMAGE>
      <SERNO>4</SERNO>
      <MuesumNumber>IM9659</MuesumNumber>
      <CATEGORY>Female, Stone,
      Standing</CATEGORY>
      <MATERIAL>Stone, Limestone</MATERIAL>
      <KEYWORDS>Female, Standing,
      Stone</KEYWORDS>
      <DESCRIPTION>Statue of female wearing
      flounced garment leaving one shoulder
      bare, hands folded, standing on flat base
      </DESCRIPTION>
      <DIMENSION>HEIGHT/LENGTH/ 36cm, tall/0.059,
      medium/0.8, small//0.4 </DIMENSION>
      <LOCATION>Khafaji</LOCATION>
      <PERIOD>Sumerian, Early Dynastic,2600
      B.C.</PERIOD>
      <STATUS>Stolen</STATUS>
      <URL>http://MySite/ImageGallery/Images/st
      anding_pic4.jpg</URL>
  </IMAGE>
</IMAGES>
```

Fig. 2 Some cases in XML case-representation

Suppose the user entered the data in Table 1 during an interactive session with the system. In the transformation of the measurement this table, the fuzzifier handles the quantitative values that need to be converted into qualitative data. Usually, we classify the artifact height into three classes: tall, medium and small. Using the membership functions, given above, the fuzzifier converts the height value 0.65 into membership grades of the respective classes: 0.88 for tall, 0.25 for medium and 0.13 for small. However, if the α-cut is

set to 0.5, then the height, in this case, is classified as tall/0.88 only.

TABLE I
AN IMAGE INSTANCE

Attribute	Value
MuseumNumber	
CATEGORY	multicasts
MATERIAL	Limestone
KEYWORDS	Female Standing
DESCRIPTION	Standing female, eyeballs of shell
HEIGHT	65
LOCATION	Tell Asmar
PERIOD/YEAR	2600
STATUS	Stolen
URL	Standing_Pic1.jpg

Once a problem instance is indexed, four additional attributes are added before it becomes a case to be stored in the case base (CB). These additional attributes are: the case number, the unusual or the interesting property. Artifacts can be characterized of having unusual properties if their heights or sizes are too small or too large, i.e. out of the usual height or size ranges. Angles or animals that take human shape and vice versa are examples of artifacts with interesting property. At the final stage a case number is assigned and the case is added to the case base (CB). Fig. 3 depicts the result of this analysis and transformation processes.

```
<IMAGE>
    <SERNO>4</SERNO>
    <MuesumNumber>IM9659</MuesumNumber>
    <CATEGORY>Female, Stone,
       Standing</CATEGORY>
    <MATERIAL>Stone, Limestone</MATERIAL>
    <KEYWORDS>Female, Standing,
       Stone</KEYWORDS>
    <DESCRIPTION>Statue of female wearing
       flounced garment leaving one shoulder
       bare, hands folded, standing on flat
       base </DESCRIPTION>
    <DIMENSION>HEIGHT/LENGTH/ 36cm, tall/0.059,
       medium/0.8, small//0.4 </DIMENSION>
    <LOCATION>Khafaji</LOCATION>
    <PERIOD>Sumerian, Early Dynastic,2600
       B.C.</PERIOD>
    <STATUS>Stolen</STATUS>
    <URL>http://MySite/ImageGallery/Images/sta
       nding_pic4.jpg</URL>
</IMAGE>
```

Fig. 3 XML case, which is the result obtained from the transformation of the data in Table I

Retrieval is an important process in case-based reasoning. Faced with a problem instance, the case based reasoning (CBR) first ranks cases in CB based on their degree of similarity with the problem instance. A similarity score that is computed by comparing each case with the problem instance quantifies this. Next, CBR retrieves the most similar cases. For improving retrieval we used a fuzzy method that combines the fuzzy terms with known qualitative attributes and uses them as keys for retrieval of similar cases. The selection of past cases that best match the present problem depends on being able to identify and evaluate relevant attributes and being able to perform simple matching between cases. Given the cases in Table II, suppose the goal is to retrieve an image similar to that described by Table II. After transformation of data in Table 1, the following problem instance is produced and added to be a new entry in the XML database.

Based on the matching attributes of the problem instance, the case retrieval can easily select the cases 1, 2, 6 and 7 from the CB to be used as bases for performance evaluation of this new problem instance. Fuzzy retrieval often results in a set of candidate cases for reasoning. The issue following fuzzy retrieval is to find the most similar case among candidates. There are several ways of finding the most similar case. In this work, we use the following algorithm (similarity measure). The similarity measure, dq, is calculated as follows:

1) $d_q = \sum_{i=1}^{n} a_i$; where n is the number of the attributes.
2) The parameter, ai, is set to -1 if the unusual-property for both the problem instance and the case has the same value; ai is set to 0 if the attribute's value for the case is equal to the attribute of the problem instance; ai is set to 0.5 if the attribute's value for the case is a wildcard. Otherwise the measure for the attribute is set to 1.
3) The similarity measure for fuzzy attributes is calculated as follows:
$$d_i = \sum_j abs(x_{ijk} - x_{ijn})$$
4) Where x_{ijk} and x_{ijn} are the grades of attribute i, class j, for cases k and n respectively.
5) The similarity measure for the case is the sum of the results obtained from (1) and (2).
$$d_c = d_q + d_i$$

Table II displays the results of applying this algorithm to the problem instance and the cases in Table 3. Uruk concludes that case 6 is, therefore, the most similar case to the problem instance and displays the associated image along with the close relative images on the screen, see Fig. 4.

TABLE II
DISTANCES BETWEEN THE PROBLEM INSTANCE OF FIG. 3 AND THE CANDIDATE CASES

Case	Fuzzy Height	Distance
1	tall/0.62, medium/0.25, small/0.13	0.85
2	tall/0.54, medium/0.9, small/0.45	0.2
6	tall/0.57, medium/0.85, small/0.42	0.09
7	tall/0.92, medium/0.15, small/0	1.38
Inst.	tall/0.59, medium/0.8, small/0.4	0

TABLE III
CASES IN SAMPLE BASE CASE (CB)

	Key Words	Descript.	Fuzzy Height	Fuzzy Price	Period/Year	Unusual Property	Status
1	Female	Standing	tall/0.62 medium/0.25 small/0.13	*	2600	*	Stolen
2	Female	Wearing garment, bare shoulder	tall/0.54 medium/0.54 small/0.45	*	2800	*	Unknown
3	Male	Bearded, long hair Wearing skirt	tall/0.3 medium/0.6 small/0.7	*	2600	*	Stolen
4	Female	Standing, wearing garment	medium/0.45	*	2600	*	Stolen
5	Male	Standing, wearing skirt, beardless	*	*	*	*	Unknown
6	Female	Wearing garment, one shoulder bare	heigh/0.57, medium/0.46 small/0.42	*	2600	*	Unknown
7	Female	Standing	heigh/0.92medium/0.15	*	*	*	Stolen
8	Male	Wearing short skirt	tall/0.45 medium/0.5 small/0.42	*	1800	*	Stolen
9	*	Standing figures	heigh/0.3 medium/0.4 small/0.7	*	1800	*	Stolen
10	Male	Standing, bearded	tall/0.9 medium/0.4	*	2800	*	Unknown

Fig. 4 The results of searching for images when the data in Table 1 are entered

Clicking on any of the retrieved images, more information on that image along with its large size will be displayed. Fig. 5 shows the result of this last action.

Fig. 5 Image along with its details

IV. THE VISUAL INTERACTIVE COMPONENT

The main components of this system are XML documents, XML database, XML processor, and Graph Manager. The XML processor, which is supported by XML parser (JAXP), has two functions: transforming the XML database into proper XML documents and vice versa. The Graph Manager, supported by the graph library Jung, is the interface module. It accepts an XML document and produces a tree-like structure that is displayed on the screen. The Graph Manager has also another task; it converts the tree-like documents to XML documents. Fig. 6 shows the interaction of these components.

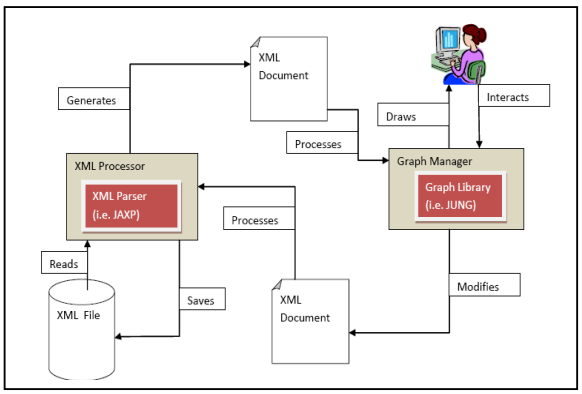

Fig. 6 The architecture of VIC

A. Visualizing an Exiting XML file

1) User chooses to import the XML file and selects the file.
2) The file address on disk is sent to XML Loader (part of XML Processor).
3) XML Loader verifies the file's structure according to the standard schema.
4) If any error is found, then exception is thrown.
5) If no error, the file loads in memory as an XML Document object.
6) Document object is sent to Plotter (Part of Graph Manager).
7) Plotter reads the Document object's contents and generates the graph by creating the corresponding vertices and connecting those using edges.
8) The graph is sent to the currently active Canvas window to be inserted and shown to the user.
9) User chooses to add/remove/modify a node.
   9.1. Receive required information/confirmation.
   9.2. Modify the Document object accordingly.
   9.3. Go to step 6.

B. Visualization of XML Documents - Screenshots and Workflow

When an XML file is visualized, VIC allows users to carry out many actions including update, delete, relocate collapse/explode nodes and reconnect a node (s) on another branch of the tree-like structure. In addition, the system allows the user to rotate the whole image. If the user clicks on a node, the color of that node will be changed from red to yellow and the associated information to that node will be displayed. The user can select a number of nodes by holding down the Shift Key while clicking on these nodes. Again, the color of those nodes will be changed to yellow and the association information will be displayed, see Fig. 7.Nodes can be collapsed to improve complex graphs' readability, see Fig. 8. When a node is collapsed, its shape will change according to the number of immediate successors it has e.g. Square if it has 4 children. Users can Collapse and Expand the nodes by right clicking on them in “Picking" mode.

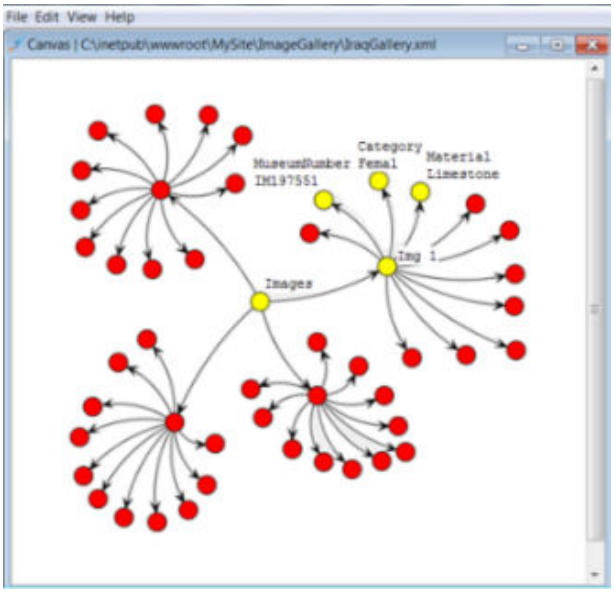

Fig. 7 Circular display of the XML document

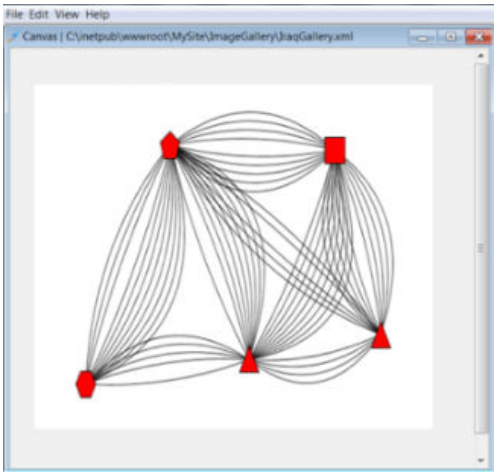

Fig. 8 Collapse of the nodes

V. Conclusion

The current research shows not only promising public domain data-centric visualization software systems running on the personal computer platform but also the effectiveness and the usefulness of such systems to the users.

In this paper we have described the Uruk system for visualization of data-centric XML collections of images. The system is based on an efficient visualization method that utilizes the JUNG software library in order to improve its capabilities. To get some insights into the functionality of Uruk, we showed some of its features using an XML document. Further research areas include the visualization and management of multiple XML documents. This is important to allow users visually moving a node (s) from one document to another.

REFERENCES

[1] Art History Virtual Library, (2008). Retrieved 10 11, 2011, from the World Wide Web Virtual Library: http://www.chart.ac.uk/vlib/images.html

[2] Photography Defence. (2010). Retrieved 10 10, 2011, from defence image database:
http://www.defenceimagedatabase.mod.uk/fotoweb/Grid.fwx.

[3] The Virtual Museum of Japanes Art. (n.d.). Retrieved 10 11, 2011, from Japan Museum Web Site: http://web-japan.org/museum/menu.html

[4] The Web Gallery of Art. (2004). Retrieved 10 11, 2011, from The Web Gallery of Art: http://www.wga.hu/index1.html

[5] The Web Gallery of Art Retrieved 5 11, 2011: http://www.wga.hu/

[6] Tourism Image Australia Gallery, (2009). Retrieved 10 11, 2011, from Tourism Australia: http://www.images.australia.com/

[7] Iraqi Museum Database. (2003, May 27). Retrieved 10 12, 2011, from Oriental Institute of the University of Chicago: http://oi.uchicago.edu/OI/IRAQ/dbfiles/Iraqdatabasehome.htm

[8] Burch, M., Diehl, S., and Weissgerber, P. Visual data mining in software archives. ACM Symposium on Software Visualization. ACM Press, pp. 37--46, 2005.

[9] Collberg, C., Kobourov, S., Nagra, J., Pitts, J., and Wampler, A system for graph-based visualization of the evolution of software. ACM Symposium on Software Visualization. ACM Press 77--86, 212, (2003)

[10] Shihab, K., Ramadhan, H. and Al-Chalabi, N. An Integrated Approach to Digital Objects Storage and Retrieval, Journal of Computer Science, 2 (9), pp. 683--689, (2006)

[11] Khalil Shihab and Doreen Ying YingSim, Development of a Visualization Tool for XML Documents, INTERNATIONAL JOURNAL OF COMPUTERS, Issue 4, Volume 4, pp. 153-160, 2010.

[12] Shihab Khalil. An Intelligent XML-Based Image System, Wseas Transactions on Computers, Vol. 5, pp. 2885-2893, 2006

[13] http://www.graphviz.org/, last accessed April 2011

[14] http://hydra3d.sourceforge.net/indexFrames.html, last accessed April 2011.