Chapter 40

Another Solution to Publish Distributed SGML/XML Documents onto the Web

Xu Jianliang, Institute of Artificial Intelligence, Zhejiang University，P.R.C. 310027, xu_jianliang@163.net

Li Shanping, Institute of Artificial Intelligence, Zhejiang University，P.R.C. 310027, shan@cs.zju.edu.cn

Ma Dan, Institute of Artificial Intelligence, Zhejiang University，P.R.C. 310027, mhj@cs.zju.edu.cn

Abstract

Publishing enterprise's on-line information is always done by making use of SGML/XML to express documents and the existing Web environment to access. One possible solution is to develop the client-side plug-in application for SGML/XML documents, but shows weakness in some extent. This paper presents another solution to create, to search, and to access distributed SGML/XML documents catering for the enterprise-wide heterogeneous environment.

Keywords: SGML/XML, distribution, Web, SGML agent

1 Introduction

It is often desirable to offer the same content both on paper and in electronic form. Technical documents, as a good example, are ideally delivered as printed manuals, in online help systems (or CD-ROM) and on the Web. Organizations that produce such service information face the same competitive challenges: how to deliver information through different media efficiently, accurately, and quickly? How to streamline processes, improve authors' productivity, reduce redundant work? How to protect sensitive information from hardware and software obsolescence?

Traditional word processors, such as MS Word, are good at producing reports, letters, etc. But their strength is also a hidden weakness as they are really limited to page-oriented documents. They use special markups, so conversion from one format to the other is non-trivial.

Only the descriptive markup language such as HTML, SGML [1] or XML [2], which describes the purpose of a document rather than its physical appearance on that page, is neutral towards media. The three languages mentioned above have their own strength and weakness:

HTML is probably the most portable markup language in the world. It is supported by over 100 million Web browsers, and is becoming the de facto standard for exchanging on-line information among people. The simplicity that makes it popular also comes at the cost of severely limiting HTML in several important respects. The most obvious are extensibility, structure and validation [3]. HTML also lacks of stability and intelligence.

In contrast to HTML, a generic SGML application is one that supports SGML language specifications of arbitrary complexity and makes possible the qualities of extensibility, structure, and validation missing from HTML. SGML makes it possible to define specific formats for specific documents, to handle large and complex documents, and to manage large information repositories. However, full SGML contains many optional features that are not needed for Web applications, and have been proven to have a high cost/benefit ratio unattractive to current vendors of Web browsers. So the SGML based Web browser is out-of-the-way.

XML is a subset of SGML allowing users to use the beneficial features of SGML easily on the Web. XML has been designed for maximum expressive power, maximum teachability, and maximum ease of implementation. The XML 1.0 has become W3C recommendation [2].

From above discussion, it is concluded that the best answer is to make use of all their advantages. That is, the information is expressed with SGML/XML and delivered via the Web (HTML application environment). One solution is based on PLUG-IN application [4]. When the Web browser loads SGML file, it launches the SGML PLUG-IN to browse it. However, this solution has following weakness:

- It can only provide document based browsing.
- It isn't able to set access rights control on a single document from the server's view.
- Such distribution doesn't own integrality and consistency, in fact, based on the whole Internet.
- It requires special Web browser, which has installed the SGML PLUG-IN application.

This paper presents another solution to manage distributed SGML/XML documents based on the enterprise-wide heterogeneous environment. It not only is able to makes the SGML/XML documents stored in the distributed databases, but enables access of them over the Web/Internet, and offers a systemic solution for the management of the SGML/XML documents. In this paper, we'll introduce the architecture of this solution and its realizing method, and discuss the main components: the SGML/XML publisher, and the database engine, the convert tool and the data directory agent of the SGML/XML agent. Since the operation method on the SGML or XML documents is similar except the parser, we illustrate the solution with SGML, and XML as the same.

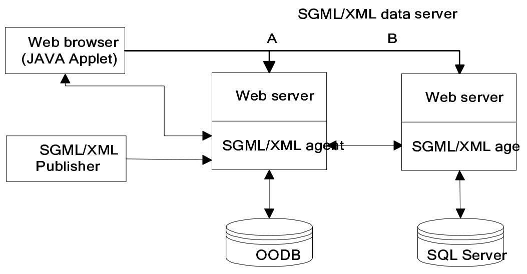


Figure 1. The systematic architecture of our solution

2 Our Solution

Fig. 1 shows the systematic architecture of our solution. It adopts "federative" architecture to organize the distributed SGML documents, which not only enables distributed storage of the SGML documents but also leaves every SGML data server independent. That is, each SGML data server is responsible to maintain a global data directory independently so that they can make up the relation of the distributed SGML documents and dynamically keep consistency of the data directory stored dispersedly in the data servers. The SGML related requests from the Web browser and the SGML publisher are accepted and handled by the SGML agent. Fig. 2 describes the architecture of the SGML agent.

In order to enable access of the SGML documents on the Web/Internet, the SGML agents should be concerted on server side to decode the SGML documents and convert them to HTML files according to relevant semantic rules. In this way, it overcomes the weakness of the PLUG-IN's solution, and assures browsing with general Web browsers. It has following main advantages:

- It is able to provide not only document based also element based browsing, and offers location-independent linkage.
- It enables distributing SGML documents enterprise-wide and making good use of the distributed database to manage data.

- It can set access rights control on a single document.
- It doesn't add any load on the client side, so it's a kind of thin-client mode.

With this solution, the SGML agent, the Web server and the convert tool have to co-operate and the requirement for the data server is relatively high. In fact, the SGML agent not only can provide the transition from SGML to HTML, but also the information in the database is able to output to other media such as CD-ROM and paper if dynamically extending the convert tool. So the reusability is really available. In addition, every the server-side operation on the SGML documents is handled by the SGML agent. It's a systemic solution.

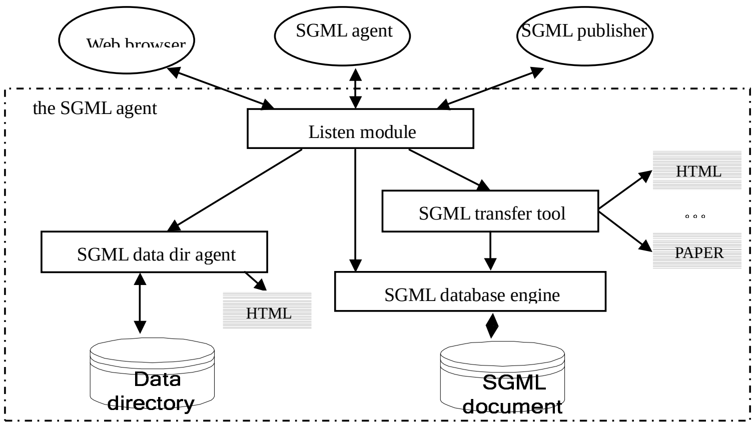


Figure 2. The architecture of the SGML agent

3 The Client Side

3.1 The Web Browser

The basic functional requirements for Web browser as a SGML browser are:

1) to search the SGML document database by keywords, and

2) to browse the content and structure of some SGML document.

The requirements entail its communication with the SGML agent. The Java-Applet is able to satisfy it, and also to cater for heterogeneous system. For example, the algorithm for browsing the document's structure and content is listed as follows:

```
if (successfully makeup the communication with the SGML agent on the same host as Web server lies){
    send the message, which contained message type of browsing and requested document name;
    wait for the SGML agent to respond;
    get the URL responded by the agent;
}
else get the URL showing the failure;
navigate the specified URL;
```

3.2 The SGML Publisher

The SGML publisher provides WYSIWYG editing for the SGML documents. Once finished, the completed document and related semantic rules may be submitted to the local or remote database via the SGML agent. To do so, it maintains two lists, one of which is SGML element based semantic-rule list and the other is editing-unit list to represent the document instance, which contains the pointer to relevant element semantic rule in rules list.

It uses the following data structure to represent the elements' semantic-rule list.

```
struct _Semantic{
        Semantic *next;                 // point to next tag's semantic
        char *gi;                       // the element identifier
        SemanticDisplay *display;       // the display semantic description
        SemanticAction *action;         // the action semantic description
};
```

4 The SGML Agent: The Realization

The SGML agent provides services for the requests from the Web browser, SGML publisher and other SGML agents, i.e., the request of Web browser accessing the content and structure of some SGML document. The SGML agent is a listener program in practice, which is consisted of the listener module, the SGML data directory agent, the SGML database engine and the convert tool. Fig. 2 shows its architecture.

4.1 The SGML Data Directory Agent

The distributed data directory should contain data dictionary and the information such as physical location of remote data and network control. Conceptually, the SGML data directory is a set of information about the SGML documents, SGML elements, SGML entities and so on in the distributed environment. Practically, the SGML data directory on every SGML data server should maintain one consistent data directory about the information of all the documents in distributed system, which assures that the user accesses the SGML documents pellucidly.

In order to satisfy the general requirements such as search and browse, it designs a table, known as GENERAL_TAB, to store the general information of the SGML documents, including the document name, authors, created date, key words, description, and its physical location, which is represented by a network address and the entry to the SGML database.

In order to satisfy intellectual access requirement, it designs another table, knows as CONTROL_TAB, to store the range of network address and according semantic rule ID. In this way, the SGML agent can identify the visitor's network address and carry out a prescriptive display intellectually. For instance, some elements' content in a document named MyCompanyInfo is only permitted visible for the staff and others information can be shared by everyone on the Internet. Suppose that the company's IP varies from 210.32.132.2 to 210.32.132.92 and there're two semantic rules, namely STYLE_DISPLAY and STYLE_NODISPLAY, the former of which displays the content mentioned above and the other not. The following shows one possible storage in the CONTROL_TAB(0-0 demotes default rule).

DOC_NAME	IP_START	IP_END	SEMANTIC_STYLE
MyCompanyInfo	0	0	STYLE_NODISPLAY
MyCompanyInfo	210.32.132.1	210.32.132.190	STYLE_DISPLAY

The algorithm of choosing semantic rule is as follows:

```
get the default ID of semantic rule;
for ( all rows of DOC_NAME == the requested document name )
     if ( the vistor's network address is between IP_START and IP_END ){
       update the semantic rule with the SEMAN_STYLE field of current row;
       break;
   }
return the right semantic rule;
```

And the executing course of the above algorithm is transparent for users and it doesn't need complex operation such as account management.

The operation such as searching, adding, deleting and altering some item of the data directory entails its share, which can be harmonized by the DBMS. However, it's the most important to dynamically keep the consistency of data directory on each SGML data server.

4.2 The SGML Database Engine

The SGML database engine provides the interface to the SGML document database. It requires interacting with the database when the SGML publisher submits the completed SGML document or the user requests to browse the content of the SGML document.

The SGML document database may be any heterogeneous database such as OODB, relative database and the inter-operation among these databases is able to be solved well with this solution. The SGML mid-format data is maintained by its DBMS but the entry to the database of the document by the SGML data directory. The storage in the SGML database is the mid-format data of the document, since they are able record all the information of the SGML document. And the benefit is to avoid re-parsing and obtain high efficiency. On finding the data stored is obsolete, it will parse the latest document automatically and decode it into the database. For example, We adopt the following data model (DDL) to represent the mid-format data when using the OODB OSCAR [6], which is developed by the Institute of Artificial Intelligence of Zhejiang University.

```
  class SGMLClass : root{
      explicit {
      char *SGMLname;          // the DTD identify of the document
      char *SGMLdoc;           // the ID in database
      EntityClass *entity;     // the entity list
      ElementClass *element;   // the element list
      ContentLink *first_tag;  // the list of the document instance
      SemanticLink **seman;    // the list of semantic rules
      };
}
```

The DDL above is able to reflect the relation between SGML objects and their logic structure. Similarly, if using relative database to store the mid-format data, it should define some tables and make up the across-reference among them.

4.3 The SGML Convert Tool

Our objective for the SGML convert tool is to offer a suite of tools to make the original SGML document output to multiplicate media such Web, CD-ROM and paper. In order to enable browse and search of the SGML document on Web/Internet, we've developed the convert tool, known as SGMLToHTML, to convert the SGML document to HTML files and put them onto the Web so that we can only use general Web browser (such as IE, Netscape) browsing the related information, content and structure of the SGML document.

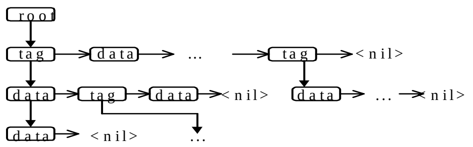


Figure 3. Resprenting SGML structure

When converting, we need travel the SGML document instance (mid-format data retrieved from the database). Its algorithm is similar with the one of the SGML publisher. Fig. 3 shows the data structure of the structure tree of the SGML document, which stores the information of SGML element's identifier and the pointer to child elements and their content in *tag,* and stores the text content of the elements in *data*.

5 Case Study

With the solution above, we've developed a prototype system. But there's a little change for simplification. Fig. 4 shows its architecture. The main difference with Fig.1 is that it replaces the "federative" data directory organization method with "central" one. In this way, we must assure that the SGML data directory server works well in the distributed environment, which is responsible to maintain the directory data and offers all services related with them. This system can be used in the Web publishing of the SGML documents. The user can just use general Web browser, such as IE and Netscape, to access the hypermedia SGML document on-line. Fig. 5 shows the appearance of the SGML document in the browser, of which the left half is the structure of the document and the right half is its content.

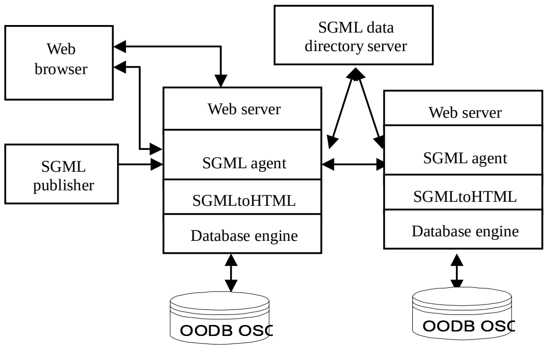
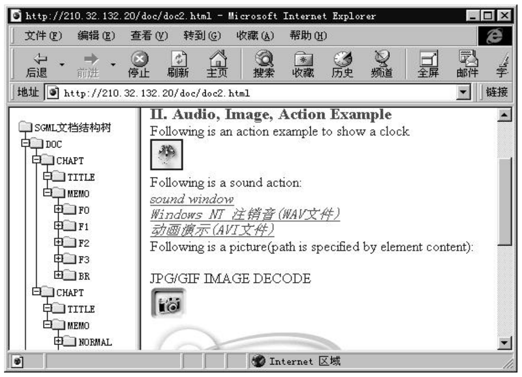


Figure 4. Systematic architecture of the prototype system


http://210.32.132.20/doc/doc2.html - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 转到(G) 收藏(A) 帮助(H)
后退 前进 停止 刷新 主页 搜索 收藏 历史 频道 全屏 邮件
地址 http://210.32.132.20/doc/doc2.html
链接
Internet 区域

SGML文档结构树
DOC
CHAPT
TITLE
MEMO
F0
F1
F2
F3
BR
CHAPT
TITLE
MEMO
NORMAL

II. Audio, Image, Action Example

Following is an action example to show a clock

Following is a sound action:

sound window

Windows NT 注销音(WAV文件)

动画演示(AVI文件)

Following is a picture(path is specified by element content):

JPG/GIF IMAGE DECODE

Figure5. the appearance of the SGML document in the browser

6 Conclusions

This solution is able to solve the exchange and share of distributed SGML/XML documents based on heterogeneous environment and enables transparent access with general Web browsers. Besides, the popularity of the Web greatly enhances its applicability.

Acknowledgment

The authors would like to thank the anonymous reviewers for their comments and suggestions. This research is supported by the National 863 program of China.

References

[1] Charles F. Goldfarb. *SGML Handbook.* Oxford University Press, ISBN 0-19-85373-9

[2] http://www.w3.org/XML/

[3] Jon Bosak. *XML, Java, and the future of the Web.* Sun Microsystems, 1997.3

[4] http://www.softquad.com/products/panorama /panvfeat.htm

[5] Shou Yucheng. *OSCAR:Object-oriented Engineering Database Management System.*Journal of CAD&CG, 1994,4