WELCOME TO THE GATHERING PLACE

Computer Networks and ISDN Systems 30 (1998) 11–21

COMPUTER NETWORKS and ISDN SYSTEMS

Structured graph format: XML metadata for describing Web site structure

Olivier Liechti [*,1], Mark J. Sifer [1], Tadao Ichikawa [1]

ISL, Hiroshima University, 1-4-1 Kagamiyama, Higashi-Hiroshima, 739 Japan

Abstract

To improve searching, filtering and processing of information on the Web, a common effort is made in the direction of Metadata, defined as "machine understandable information about Web resources or other things". In particular, the eXtensible Markup Language (XML) aims at providing a common syntax to emerging Metadata formats. With this idea, we propose the Structured Graph Format (SGF), an XML compliant markup language based on structured graphs, for capturing Web sites' structure. We also present SGMapper, a client-side tool, which aims to facilitate navigation in large Web sites by generating highly interactive site maps using SGF Metadata.

Keywords: Metadata; Site maps; Structured graphs; XML

1. Introduction

Literally meaning "information about data", Metadata is defined by the **World Wide Web Consortium** [2] (W3C) as "**machine understandable information about Web resources or other things**" [3]. The motivation for a current joint effort of major companies and organizations is to make better use of WWW information, which today lacks structure and typing. This could be achieved by defining standard ways to add the "magic glue", that will allow automatic searching, filtering and processing of data transiting on intra- and inter-networks.

A common foundation for emerging Metadata formats is the **eXtensible Markup Language** [4] (XML) developed by the W3C, which provides a common syntax for a large number of Metadata formats. As discussed in [2], different applications of XML are envisaged and classified into four broad categories: (1) applications that require the Web client to mediate between two or more heterogeneous databases, (2) applications that attempt to distribute a significant proportion of the processing load from the Web server to the Web client, (3) applications that require the Web client to present different views of the same data to different users, and (4) applications in which intelligent Web agents attempt to tailor information discovery to the needs of individual users. **CDF** [5],

* Corresponding author.
[1] E-mail: {olivier,marks,ichikawa}@isl.hiroshima-u.ac.jp
[2] http://www.w3.org
[3] http://www.w3.org/Metadata/
[4] http://www.w3.org/TR/WD-xml-lang, http://www.w3.org/XML, http://www.sil.org/sgml/xml.html, http://www.ucc.ie/xml/
[5] http://www.microsoft.com/standards/cdf.htm

0169-7552/98/$19.00
PII S0169-7552(98)00039-7

DOM [6], **MCF** [7] and **OSD** [8] are some of the numerous acronyms for recently proposed Metadata formats. **WebCollections** [9] are an application of XML for describing the properties of some object, e.g. the properties of a Web document. Many other proposals have been made, in the mathematical, chemical, commercial application domains for example.

In this paper, we present the Structured Graph Format (SGF), for describing Web site structure. We also present SGMapper, a client side tool, which uses SGF Metadata to generate interactive site maps. We describe in detail how SGMapper cooperates with a standard Web browser and how any Web server can easily be set up to provide SGF Metadata. We illustrate one possible use of SGF, which could however been applied in many other settings.

The problem addressed by SGMapper is to facilitate navigation in hyperspace. When users navigate large sites by using hyperlinks only, they have difficulty maintaining context [4,5]. Many approaches to this problem try to provide some kind of overview in combination with a detailed portion of the site. Fisheye lenses are a method for browsing large graphs [7], allocating a large proportion of screen space to a restricted part of the graph and squeezing the rest into the remaining area. Fisheye lensing has been combined with zooming and applied to Web navigation in [3]. Related non-linear display methods for trees are hyperbolic maps [10] and perspective walls [12]. Another approach used by WebToc [14] and Nif-T-Nav [9] is to display the site structure as a hierarchy, which can be navigated like a file system. With the HyperG system [1], a site author explicitly designs a hierarchy of nodes in addition to associative links. An advantage of this is, that both a hierarchy view and a local view of the surrounding associative network are available.

SGF is based on structured graphs [16], originally developed in the settings of software engineering and recently applied to multimedia data modeling [11]. Structured graphs were designed to allow scalable browsing and editing of very large graphs and therefore may be useful in the context of the Web. They distinguish between hierarchical and associative links, where hierarchical links are intended to capture aggregation relationships. In addition to the explicit associative links, implicit associative links are generated to show associations between sub-hierarchies. This provides for a close integration of the hierarchy and network views, allowing some simple visual queries to be made.

The paper starts with a simple example, showing how the structure of a Web site can be described with a hierarchy and a network. The notion of implicit associative link is introduced. At the same time, navigation of the example site using SGMapper is presented. SGF is introduced with the encoding of the example site structure. Its specification is then given in the form of its Document Type Description (DTD). Some implementation and integration issues are exposed. Final conclusions are given.

2. The SGLab example

As a first step, we present the example of a fictitious Web site. While showing, how the structure of this site can be described with a hierarchy and a network, we explain how the SGMapper browsing assistant can be used to explore it. The example also introduces the notion of implicit associative links and their use in answering simple visual queries. Finally, the SGF Metadata describing the site is given.

2.1. Hierarchical organization of the site

The SGLab Web site provides information about an imaginary computer science institute. This information can be placed in a hierarchy reflecting aggregation relationships, as shown in Fig. 1a. When complex information is organized in a hierarchy, it is easier to explore [13]. By making choices when moving down the hierarchy, the user restricts the amount of information considered worth examining. For example, if the user is looking for Professor Smith, he/she will most probably follow the path: SGLab – Members – Staff – Pr. Smith, considering only 4 among the 20 nodes of the site.

The upper frame in Fig. 2a and b illustrates how the site hierarchy, generated on the basis of SGF Metadata sent by the server, is displayed in

[6] http://www.w3.org/TR/WD-DOM/
[7] http://www.textuality.com/mcf/NOTE-MCF-XML.html
[8] http://www.w3.org/TR/NOTE-OSD.html
[9] http://www.w3.org/TR/NOTE-XMLsubmit.html

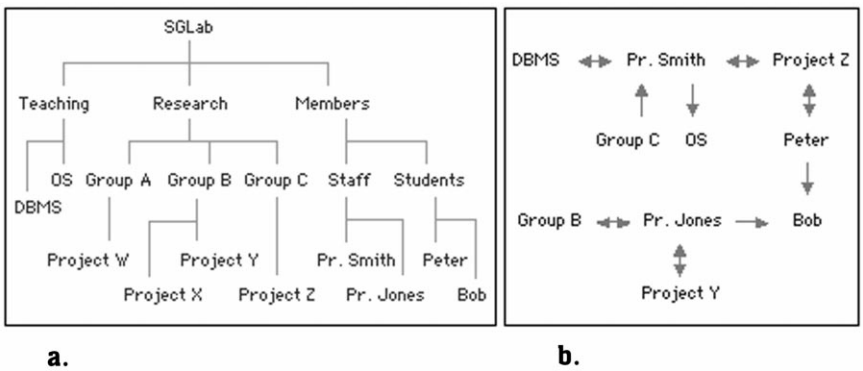


Fig. 1. Hierarchical and network organizations of the SGLab Web Site.

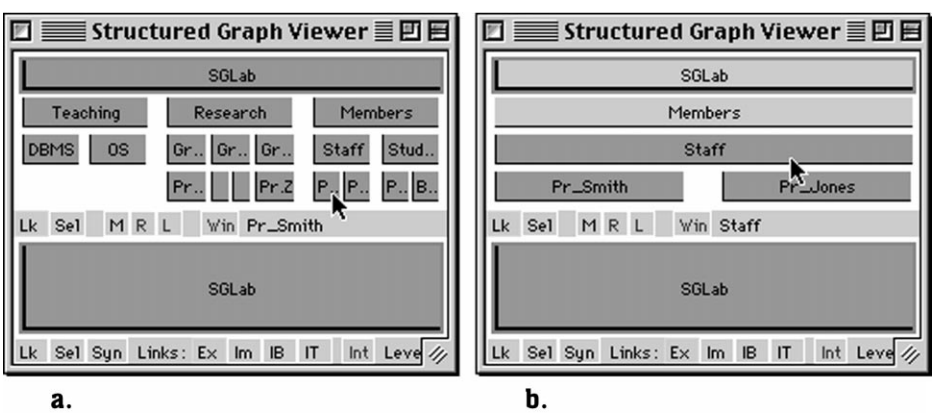


Fig. 2. Overview and context information of the hierarchy.

SGMapper. The user can click on a node to select it, or shift-click on a node to zoom in the hierarchy. In Fig. 2b, the user has zoomed on the Staff node by shift-clicking it and can see only the ancestors and descendants of this node. These two user operations determine the nature of the information displayed in a window: in the first case (Fig. 2a), the user can see an overview of the hierarchy without seeing labels for every node, while in the second case (Fig. 2b) details for only a portion of the structure are shown. Both views are useful and it is possible to switch from one to the other in a single operation. Moreover, the user can open multiple windows, displaying the hierarchy with different focus points. When a node is selected, visual feedback is given in all windows. This helps the user maintain context while getting details. Finally, because SGMapper is designed to cooperate with a standard Web browser, if the user double-clicks on a node, the corresponding URL is automatically downloaded and displayed in the browser.

2.2. Network organization of the site

We have said, the site hierarchy is defined by aggregation relationships. There is another type of relationship, the associative relationships which define a network. For example, this is the case when a lab member is associated with a project, as is Pr. Jones with Project Y. Figure 1b shows the associative links of the SGLab Web site.

Figure 3 shows a portion of the site network reachable from Pr. Smith displayed in SGMapper. The upper frame represents the hierarchy and the lower one represents the network. The user has selected Pr. Smith in the hierarchy frame. Consequently, Pr. Smith appears in the center of the network frame, where each node has on its left side

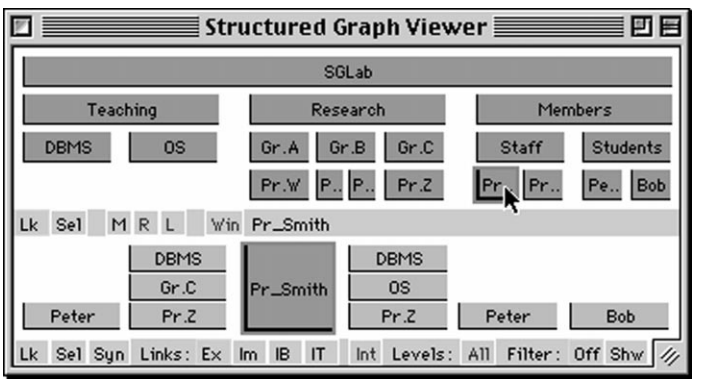


Fig. 3. Network of associative links for the node selected in the hierarchy.

the associative link sources and on its right side the associative link destinations. For example, DBMS is on both the left and right sides of Pr. Smith, because there are two opposing links joining these nodes. The user can explore the network either by clicking or shift-clicking a node. The resulting behavior is the same as in the hierarchy frame, that is either overview or detailed information of the structure is displayed, or both at the same time if multiple windows are opened.

2.3. Implicit associative links

Up to now, we have considered two types of links: aggregation links, which define a hierarchy, and explicit associative links, which define a network. We now introduce implicit associative links, which are derived from the hierarchy and the network. Implicit associative links capture relations between sub-hierarchies. As shown in Fig. 4, there exists an implicit associative link between the sub-hierarchies P and Q, when there is at least one explicit associative link between the elements s and d, respectively in P and Q. Formal definitions are given in Appendix A. An example of an implicit associative link in the SGLab site is: from Group C to Students, where P = Group C, Q = Students, s = Project Z and d = Peter.

SGMapper provides a number of features, which allow users to select portions of the structure and to determine which explicit/implicit associations between them are displayed. Space prevents us from describing these functions in detail. However, we would like to illustrate their ability to answer simple visual queries. Consider the research groups A, B

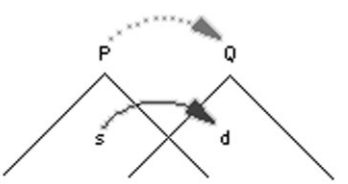


Fig. 4. Implicit associative link.

and C and the member categories Staff and Students. The user could be interested to find out, what kind of members are associated with each group. Figure 5a to c shows, how the user could answer this question with SGMapper. The user needs to select the Member's sub-hierarchy and to successively click on each of the three Research groups to see the associations. First, group A is selected and no association with any member is seen. Second, the user clicks on group B, as shown in Fig. 5a. There is an (implicit) link between Group B and Staff, but not between Group B and Students. Third, the user clicks on Group C, as shown in Fig. 5b. There are (implicit) links between Group C and Staff, and between Group C and Students. The interpretation of the three views gives the answer: both students and staff members are associated with group C, only staff members are associated with group B, and no one is associated with group A. In Fig. 5c, another function of the viewer is used to further restrict the displayed associations.

2.4. SGLab Web site description with SGF

We have shown, how the structure of a site can be described with hierarchical and associative links.

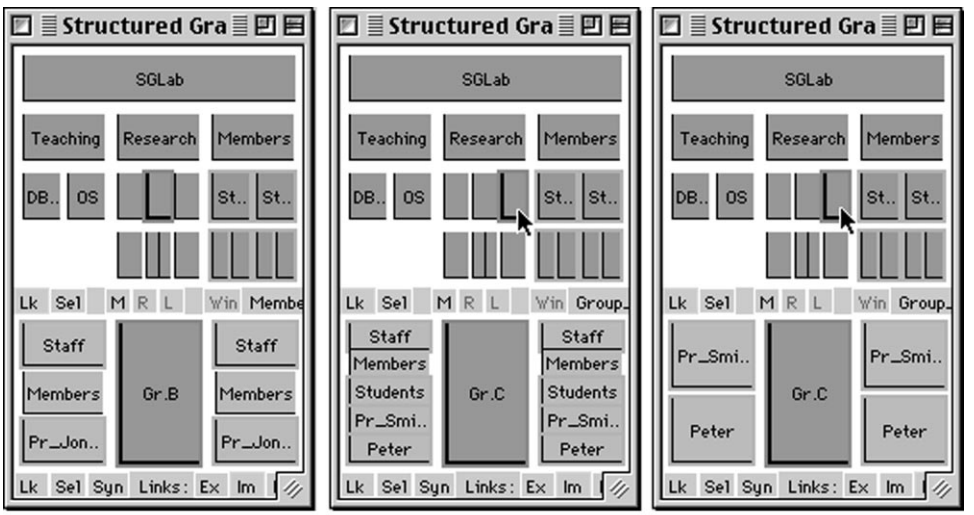


Fig. 5. Implit associative links allow simple visual queries to be answered.

We now introduce SGF, a format for encoding this information. Table 1 shows a portion of such a file, describing the example site:

- the site structure is captured as a **structured graph**, defined between the <STRUCTUREDGRAPH> and the </STRUCTUREDGRAPH> tags,
- the **set of nodes** is defined between the <NODES> and the </NODES> tags,
- the **set of hierarchical links** is defined between the <HIERARCHY> and </HIERARCHY> tags,
- the **set of associative links** is defined between the <NETWORK> and </NETWORK> tags,
- each **node** is defined by NODEID and LABEL attributes, the URL is defined in an optional <SGATT> sub-element between <NODE> and </NODE> tags,
- each **link** is defined by SOURCE and DEST attributes matching one of the NODEID attributes, within <LINK> and </LINK> tags

3. SGF specification

The previous example has presented a possible way to describe site structure. We have also shown a partial SGF document. We now present the SGF specification, in the form of its XML Document Type Description (DTD). As raised in the introduction, the need for Metadata on the Web is becoming widely recognized, to make huge quantities of information available not only for human browsing, but also for machine understanding. Many of the emerging formats use the eXtensible Markup Language (XML) as a common syntactic layer. An XML document contains text and nested tags defining elements and attributes. A DTD defines the constraints for a document class.

Table 2 gives the SGF DTD. Without explaining in detail the syntax of an **XML DTD** [10], we describe the SGF validity constraints with reference to the related XML keywords given in parenthesis:

- the root of an SGF document must be a STRUCTUREDGRAPH element,
- a STRUCTUREDGRAPH element can have an arbitrary number of SGATT sub-elements, followed by a NODES, a HIERARCHY and a NETWORK sub-element (SGATT*, NODES, HIERARCHY, NETWORK),
- the NODES element can have an arbitrary number of NODE elements (NODE*),
- the HIERARCHY element can have an arbitrary number of LINK elements (LINK*),
- the NETWORK element can have an arbitrary number of LINK elements (LINK*),

[10] http://www.w3.org/TR/WD-xml-lang

Table 1
SGF description of SGLab Web site

```
<!DOCTYPE STRUCTUREDGRAPH SYSTEM "SGF.dtd">d">
<STRUCTUREDGRAPH>PH>
 <NODES>
  <NODE NODEID="N001" LABEL="SGLab_Home_Page">
   <SGATT NAME="URL" VALUE="http://www.sglab.edu"/>
  </NODE>
  <NODE NODEID="N002" LABEL="Teaching">
   <SGATT NAME="URL" VALUE="http://www.sglab.edu/teaching.html"/>
  </NODE>
  <NODE NODEID="N003" LABEL="Research">
   <SGATT NAME="URL" VALUE="http://www.sglab.edu/research.html"/>
  </NODE>
[...]
  <NODE NODEID="N016" LABEL="Project_Z">
   <SGATT NAME="URL" VALUE="http://www.sglab.edu/projects/project_z.html"/>
  </NODE>
  <NODE NODEID="N019" LABEL="Peter">
   <SGATT NAME="URL" VALUE="http://www.sglab.edu/p̃eter"/>
  </NODE>
 </NODES>
 <HIERARCHY>
  <LINK SOURCE="N001" DEST="N002"> </LINK>
[...]
 </HIERARCHY>
 <NETWORK>
  <LINK SOURCE="N019" DEST="N020"> </LINK>
[...]
 </NETWORK>
</STRUCTUREDGRAPH>
```

- a `NODE` element must (`#REQUIRED`) have a unique identifier (`ID`) `NODEID` attribute,
- a `NODE` element can (`#IMPLIED`) have a `LABEL` attribute,
- a `NODE` element can have an arbitrary number of `SGATT` sub-elements (`SGATT*`),
- a `LINK` element must (`#REQUIRED`) have a `SOURCE` attribute, for which there must exist a corresponding unique id attribute (`#IDREF`),
- a `LINK` element must (`#REQUIRED`) have a `DEST` attribute, for which there must exist a corresponding unique id attribute (`#IDREF`),
- a `LINK` element can (`#IMPLIED`) have a `LABEL` attribute,
- a `LINK` element can have an arbitrary number of `SGATT` sub-elements (`SGATT*`),
- a `SGATT` element must (`#REQUIRED`) have two attributes, `NAME` and `VALUE`.

`SGATT` elements make SGF extensible, allowing the addition of application specific information. SGMapper, for example, uses `SGATT` in `NODE` elements to store their corresponding URL.

4. Implementation and integration issues

We have shown, how the SGMapper browsing assistant can be used to help user navigation of a site providing SGF Metadata. We have also explained, how SGF Metadata is encoded in documents satisfying the validity constraints dictated by the SGF DTD. We now discuss some implementation issues of SGMapper, in particular its inter operability with a standard browser. We also explain, how any Web server can be setup to provide SGF information.

Table 2
The SGF DTD

```
<!- SGF DTD ->
<!- a SG is defined by 3 sets: nodes, hierarchical and associative links ->
<!ELEMENT STRUCTUREDGRAPH (SGATT*,NODES,HIERARCHY,NETWORK)>
<!- the set of nodes ->
<!ELEMENT NODES (NODE*)>
<!- the set of hierarchical links ->
<!ELEMENT HIERARCHY (LINK*)>
<!- the set of associative links ->
<!ELEMENT NETWORK (LINK*)>
<!- a node ->
<!ELEMENT NODE (SGATT*)>
<!ATTLIST NODE
 NODEID ID #REQUIRED
 LABEL CDATA #IMPLIED >
<!- a link between 2 nodes ->
<!ELEMENT LINK (SGATT*)>
<!ATTLIST LINK
 SOURCE IDREF #REQUIRED
 DEST IDREF #REQUIRED
 LABEL CDATA #IMPLIED >
<!- SGATT allows definition of application specific attributes ->
<!ELEMENT SGATT EMPTY>
<!ATTLIST SGATT
 NAME CDATA #REQUIRED
 VALUE CDATA #REQUIRED>
```

4.1. SGMapper implementation

On the client side, the system allowing a user to navigate Web sites with SGMapper integrates the following components: a standard Web browser (Netscape and Microsoft Internet Explorer are supported), a spy program and an interactive site viewer. The browser is used to access information on the Web in a completely standard way. The viewer is capable of parsing SGF documents, of visualizing the structure of the site. The spy provides integration mechanisms between them. A two-way communication channel and a monitoring of the browser are needed for the following reasons:

- when an SGF document is fetched by the browser, it needs to be automatically forwarded to the viewer,
- when the user double-clicks on a node in the viewer, the related URL should be downloaded by the browser,
- when a user navigates a site with the browser and accesses a page, visual feedback should be given in the viewer.

Standard Web browsers support some level of inter operability. Both **Netscape**[11] and **Microsoft Internet Explorer**[12] have defined an client API based the same event suite defined by **SpyGlass**[13]. Using platform dependant inter process communication (Apple Events on MacOS, DDE or OLE on Windows), it is possible to exchange messages with the browser. The following have been used in the implementation of the spy program:

- `RegisterViewer` is sent by the spy to the browser, to declare its intention to handle SGF files (according to this, every time a hyperlink pointing towards a file of SGF MIME type is followed, the file is downloaded on the client computer and opened by the spy),
- `OpenURL` is sent from the spy and starts a download and display of the specified URL by the browser,

[11] http://home.netscape.com/newsref/std/
[12] http://support.microsoft.com/support/kb/articles/Q160/9/57.asp
[13] http://www.spyglass.com

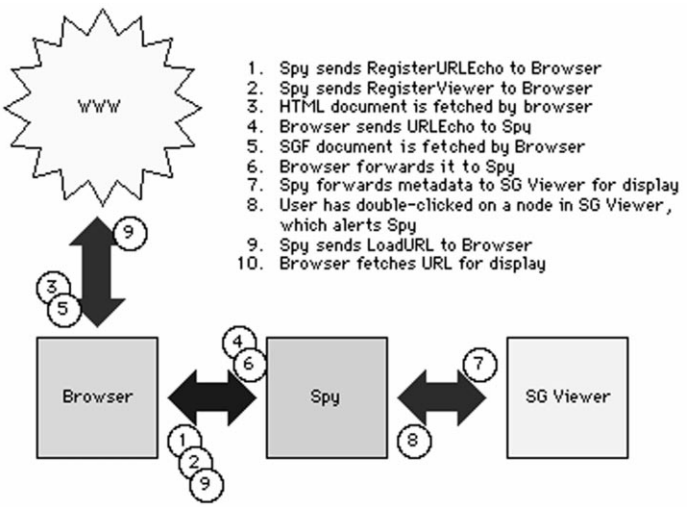


Fig. 6. Client side integration.

- `RegisterURLEcho` is sent by the spy to the browser and causes the browser to send `URLEcho` messages to the spy,
- `URLEcho` messages are sent to the registered spy every time a URL is downloaded in the browser, giving the reference of this URL and its MIME type.

The overall behavior of the system is illustrated in Fig. 6: at initialization time, the spy registers both to be alerted when URLs are fetched by the browser (1) and to handle SGF files (2). When a HTML resource is downloaded from the Web (3), the browser alerts the spy (4). When the resource MIME type is SGF (5), then the browser stores it on disk and sends a request to the spy to open it (6). The spy can then forward the Metadata to the Viewer, which can display it (7). When the user double-clicks on a node in the viewer, this alerts the spy (8), which asks the browser to download the corresponding URL (9). The resource is finally fetched and displayed in the browser (10).

At the time of writing, the implementation of this architecture has been almost completed, though some effort is still needed to achieve full inter operability between the viewer and the browser. Both the spy and the viewer have been implemented on MacOS, the former in C++ and the latter in Java. Communication between the spy and the browser is done via Apple Events, communication between the spy and the viewer is done over TCP/IP. The viewer uses **Microsoft msxml** [15] **XML parser**[14]. There should not be any major obstacles in porting the code to the Windows platform, as only the communication between the spy and the browser would need to be rewritten, using Windows based IPC (DDE for example).

4.2. Serving SGF metadata

SGMapper works with any Web server, the only restriction being, that it serves SGF Metadata. There are two requirements: (1) an SGF document must exist on the server and must be accessible via at least one hyperlink and (2) when sending this document to a client, the server must explicitly specify, that it is an SGF document.

Ideally when a visitor arrives on a site, not necessary via the home page, he/she should be able to get the Metadata quickly and easily, that is by having to follow a minimal number of links. One possibility, as shown in Fig. 7, is to have in every document a link to a unique HTML document (map.html), that gives access to an SGF document (map.sgf). This indirection for example allows a traditional site map to be provided together with some information about the Metadata format and available clients.

[14] http://www.microsoft.com/standards/xml/xmlparse.htm

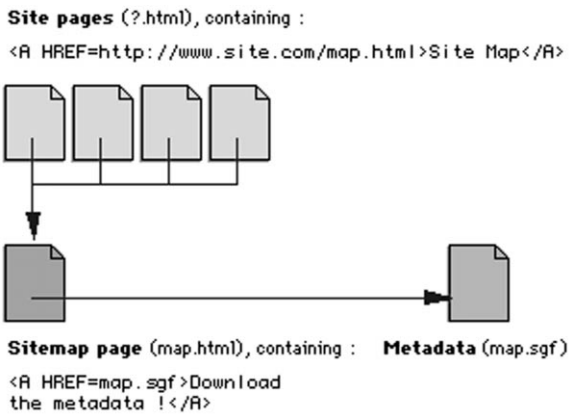


Fig. 7. Giving access to the Metadata on the Web server.

The easiest way to satisfy the second requirement is to rely on **MIME types** [15], which are a standard way of determining the content of a file sent across TCP/IP networks. For this reason, we have defined a MIME type for SGF files, namely "text/x-sgf". Web servers can easily be configured to send particular MIME types, generally by adding a line in a file called "mime.types" or the like, which defines a mapping between MIME types and a file extensions.

Finally the question of how to build the SGF document capturing the site structure remains. A straightforward method is to manually encode the information using a text editor. A more realistic approach would be to create or adapt a site design tool, such as **NetObject Fusion** [16], **SiteMill** [17] or **Front Page** [18], which could automatically generate an SGF description of the site on the basis of relationships defined by the designer. Another approach would be to generate the Metadata by parsing the documents of the site. This process would be easier, if the HTML documents contained some extra information about the links, as proposed in [8].

[15] http://www.cis.ohio-state.edu/hypertext/faq/usenet/mail/mime-faq/top.html
[16] http://www.netobjects.com/html/products.html
[17] http://www.adobe.com/prodindex/pagemill/siteben.html
[18] http://www.microsoft.com/frontpage/

5. Conclusion

The motivation for our work is based on our belief, that Metadata describing the structure of Web sites is useful. For this reason, we have proposed SGF, which allows the encoding of a site using a Structured Graph. To demonstrate, we have presented SGMapper, which uses SGF information to improve user navigation in a site.

Benefits of this system include (1) efficiency, as the structure is downloaded once, in a single transfer, permitting exploration of the site structure without sending further requests to the server, (2) help for navigation, as both context and detail information about a site are given to the user by multiple windows, and (3) the ability to answer visual queries, taking advantage of implicit associative links defined by the Structured Graph formalism.

SGMapper is only one of the possible uses of SGF Metadata. Because SGF is a public format, any client application could request SGF documents from a Web servers and process them. Software agents could take advantage of this, by regularly checking the evolution of the structure of a site and initiate appropriate actions. For example, think of an agent, which would notify you every time a new paper is written by your favourite author. Altogether, SGF illustrates three of the envisioned use of XML: (1) applications that attempt to distribute a significant proportion of the processing load from the Web server to the Web client, (2) applications that require the Web client to present different views of the same data to different users, and (3) applications in which intelligent Web agents attempt to tailor information discovery to the needs of individual users.

A first step in our future work is to complete and enhance the integration of SGMapper with Web browsers. Beyond this, we would like to explore the use of other capabilities of structured graphs [17], such as support for links having multiple destinations and sources, link hierarchies and link types. We have started to implement a robot, which explores a Web site and generates its SGF structure description using a set of rules to distinguish between hierarchical and associative links. We have also started an extension of the system, which would support storing the user's browsing history in SGF. The associated Structured Graph would grow dynamically as the user would

visit different sites, either by using SGF Metadata if provided, or by infering the nature of the links with a similar set of rules used by the robot. Format specification, demonstrations and progress reports will be published under **SGF home page** [19].

Acknowledgments

The first author was supported with a Japanese Government Monbusho Scholarship and the second author by the Japan Society for Promotion of Science with a fellowship.

Appendix A

This appendix describes a condensed version of structured graphs. Unfortunately space does not allow definitions for the standard mathematics used here to be included. However, they can be found in [6].

Definition A.1

A condensed structured graph is the tuple (<=, network) where:
<=: is a finite ordering relation on nodes: **P** Node
network: Node -> **P** Node

Definition A.2

Let G be a condensed structured graph (<=, net). Let the abstraction operator A map from condensed structured graphs to condensed structured graphs. Then A(G) = (<=, net′) where:
For all p,q: nodes such as (p,q) in net′ if and only if there exists (s,d) in net such as $p <= s \wedge p \parallel d \wedge q <= d \wedge q \parallel s$

The abstraction operator takes a whole structured graph and adds implicit links represented as node pairs to it. The rule for adding an implicit link can be summarized as: a node p is an implicit link source when it is above a node s which is an explicit link source and the link destination d is outside p, see Fig. 5. There is a corresponding requirement for the link destination.

[19] http://www.isl.hiroshima-u.ac.jp/projects/SGF/

References

[1] K. Andrews, Browsing, building and beholding cyberspace, new approaches to the navigation, construction, and visualisation of hypermedia on the Internet, Doctoral dissertation, Graz University of Technology, September 1996

[2] J. Bosak, XML, Java, and the future of the Web, Sun Microsystems, October 1997, available on the Web at http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.html

[3] G. Collaud, J. Dill, C.V. Jones and P. Tan, The continuously zoomed Web — a graphical navigation aid for WWW, in: *Late Breaking Hot Topic in conjunction with Visualization'96*, October 27–November 1, San Francisco, California, 1996

[4] J. Conklin, Hypertext: an introduction and survey, *IEEE Computer*, 20(9): 17–41, 1987.

[5] D.W. Edwards and L. Hardman, *Lost in Hyperspace: Cognitive Mapping and Navigation in a Hypertext Environment*, Intellect Books, Oxford, 1989.

[6] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, Cambridge University Press, 1990.

[7] G. W. Furnas, Generalized fisheye views, in: *CHI'86 Proceedings*, Boston, MA, pp. 16–34.

[8] D. Glazman, Navigation through LINK elements in HTML, 15 June 1997, available on the Web at http://www.w3.org/TR/NOTE-link

[9] K.L. Jones, NIF-T-NAV: a hierarchical navigator for WWW pages, in: *Proc. of the 5th International World Wide Web Conference*, May 1996, Paris, France, available at http://www5conf.inria.fr/fich_html/papers/P39/Overview.html

[10] J. Lamping, R. Rao and P. Pirolli, A focus+context technique based on hyperbolic geometry for visualizing large hierarchies, in: *Proc. CHI'95*, Denver, Colorado, *Pub. ACM*, pp. 401–408.

[11] D.B. Lowe, A. Ginige, M. Sifer and J. Potter, The Matilda Data Model and its implications, in: *Proc. of the 3rd International Conference on Multimedia Modeling*, Toulouse, November 1996.

[12] J.D. Mackinlay, S. Card and G. Robertson, Perspective wall: detail and context smoothly integrated, in: *Proc. of the ACM SIGCHI'91 Conference on Human Factors in Computing Systems*, New Orleans, LA, April 1991, pp. 173–179.

[13] S. Mukherjea, J.D. Foley, and S. Hudson, Visualizing complex hypermedia networks through multiple hierarchies, in: *Proc. of the ACM SIGCHI'95*, Denver, Colorado, USA, May 1995, available at http://www.acm.org/sigchi/chi95/proceedings/papers/sm_bdy.htm

[14] D.A. Nation, C. Plaisant, G. Marchionini, and A. Komlodi, Visualizing Websites using a hierarchical table of contents browser: WebTOC, in: *Proc. of the 3rd Conference on Human Factors and the Web*, Denver, Colorado, June 1997.

[15] J. Paoli, D. Schach, C. Lovett, A. Layman, and I. Cseri, Building XML parsers for Microsoft's IE4, *World Wide Web Journal (W3J)*, II(4), Fall 1997.

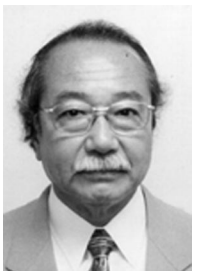
[16] M. Sifer and J. Potter, Structured graphs: a visual formalism

for scalable graph based CASE tools, *Australian Computer Journal*, 28(1): 13–26, February 1996. Errrata published in 28(2): 71, May 1996.

[17] M. Sifer, Structured graphs: a visual formalism for scalable graph based tools and its application to software structured analysis, Ph.D. thesis, University of Technology Sydney, 1996.

Olivier Liechti received his Diploma in Computer Science at the University of Fribourg, Switzerland in 1995. He has been working for several years at Prisme Informatique in Switzerland, where he was involved in DBMS and intranet related projects. He is now a PhD student at the Information System Laboratory in Hiroshima University, Japan. His main research interests are in Web technologies, CSCW systems, mobile and ubiquitous computing environments.

Mark Sifer is presently a JSPS research fellow with the Faculty of Engineering at Hiroshima University. Previously he was a Lecturer at the University of Technology Sydney, where he also completed a PhD. He has also worked at a range of companies and organisations as a software developer. His main research interest is in graph abstraction models and their application to the design of graph based software tools.



Tadao Ichikawa graduated from Waseda University also receiving his Doctor of Engineering degree from Waseda University. Prior to his professorship at Hiroshima University, he worked at the Research and Development Laboratory of Kokudai Denshin Denwa Co., Ltd. (KDD) in Tokyo. He moved to Hiroshima University in 1979, where he is responsible for research and education in information and computer sciences as Professor of the Information Systems Laboratory. He initiated the IEEE Symposium on Visual Languages and the IEEE International Conference on Multimedia Computing and Systems in 1984 and 1994, respectively. At the IEEE (Institute of Electrical and Electronics Engineers) Computer Society, he also founded the Technical Committee on Multimedia Computing and the Task Force on YUFORIC (Youth Forum in Computer Science and Engineering)in 1992 and 1995, respectively. He is on the editorial board of the IEEE Transactions on Knowledge and Data Engineering and the International Journal on Visual Languages and Computing. He currently serves as Board of Govenors at the IEEE Computer Society. He is an IEEE Fellow.