World Wide Web 2 (1999) 191–208

The SGF metadata framework and its support for social awareness on the World Wide Web

Olivier Liechti, Mark Sifer and Tadao Ichikawa

Information Systems Laboratory, Faculty of Engineering, Hiroshima University, 1-4-1 Kagamiyama, Higashi-Hiroshima 739, Japan
E-mail: olivier@isl.hiroshima-u.ac.jp

The widespread use of metadata is transforming the WWW into an information space that can be accessed not only by humans, but also by software agents. In this article, one application for metadata is more closely examined: the description of Web sites structures in a machine understandable way. The Structured Graph Format (SGF) is introduced as an XML-based format supporting the description of Web spaces as structured graphs. The SGF framework, built around this format specification, is then described. This integrated and extensible set of software components supports the generation, the distribution and the processing of SGF metadata. Three approaches to the problem of generating SGF metadata are compared and highlight a tradeoff between quality and cost. SGF consumers are then presented as components that process the metadata for some purpose. An SGF consumer that uses the metadata to dynamically generate interactive site maps is presented. The discussion then argues for the need to increase social awareness on the WWW. In other words, it raises the issue of monitoring the activity occurring within Web sites. The notion of awareness is first introduced and situated in the context of Computer Supported Cooperative Work (CSCW). Different ways to apply awareness to the WWW are then reviewed. Finally, the SGF framework is described as a valuable foundation for building awareness systems on the Web, with two main advantages. First, because SGF metadata supports the definition of regions within a Web site, at different granularities, it ensures the scalability of monitoring systems. It thus gives users of these systems a very flexible way to define regions of interest and to monitor activity in more meaningful ways. Second, the site maps generated on the basis of SGF metadata provide an efficient way to represent the activity occurring within the monitored site. These explicit representations, which are useful to analyze activity, are contrasted with abstract representations, which are useful to maintain peripheral awareness about ongoing activity on the Web.

1. Introduction

The World Wide Web was originally developed to provide a lightweight mechanism for sharing electronic documents within distributed workgroups. Open protocols were proposed for the authoring and the transfer of hypertext documents, with an emphasis put on simplicity. Hence, the WWW model has its shortcomings and has often been criticized by the hypertext research community on that basis. As reported in [Nürnberg and Ashman 1999], an argument is that the WWW does not qualify as a true hypermedia system, because it does not consider structural entities, in particular hyperlinks, as first-class citizens. The fact that WWW links cannot be manipulated (viewed, searched, etc.) on their own highlights this problem. Nevertheless, many recognize the WWW as the most successful hypertext system ever and agree that simplicity has been a critical factor for its wide adoption and tremendous growth.

Unfortunately, because only few constraints are imposed on Web authors, the resulting information space is characterized by heterogeneity and lack of structure. This lack of structure is observed both *within* documents (HTML does not capture semantics) and *among* documents (relationships between them are not explicitly defined). As a result, the automatic processing of Web resources has remained very difficult. This causes problems in many situations and considerably hinders the benefits that could be made out of a huge quantity of information and services. For example, finding relevant information on the Web is often tedious, as search engines are generally limited to the retrieval of keywords within indexed documents. While it is possible to retrieve the documents *containing* the words "Tim Berners-Lee" and "WWW", it is not possible to retrieve the documents *written by* "Tim Berners-Lee" *about* the "WWW". Electronic commerce is a particular application domain that demands the provision of semantically structured resources, in order to efficiently and securely conduct on-line business transactions.

For these reasons, a global effort is currently made to augment the WWW with *metadata*, defined by the World Wide Web Consortium (W3C) as *machine understandable information about Web resources and other things.* The idea is to transform a document space that was designed to be browsed by humans into an information space that can also be efficiently explored and analyzed by software agents. New languages, protocols and architectures are developed with this aim. The envisioned applications include resource cataloging and retrieval, information filtering, trust and security management and many others. An important element in the emerging metadata infrastructure is the already popular eXtensible Markup Language (XML) [Bosak 1997], which provides a common syntax for all domain-specific metadata formats (XML is a markup language for defining markup languages). Unlike HTML, XML provides an extensible set of markup tags that are used to capture the semantic structure of documents. A Web browser can determine how it should render HTML documents, but does not understand them. On the contrary, the information in XML

documents (e.g., price lists, bibliographies or job ads) can be analyzed and reasoned about by software components. This permits, for example, the implementation of agents that automatically retrieve cheap products by visiting Web sites on the user's behalf [Rajaraman and Norvig 1998].

Existing applications exhibit three common usage patterns for metadata. In the first case, the metadata describes the *content of documents*, i.e., their internal structure. MathML [W3C 1998a], SMIL [W3C 1998b] and XML-EDI [Webber 1998] are only three of the many XML-based formats that are used to encode, respectively, mathematical formulas, multimedia presentations and business transactions. In the second case, metadata is used to describe *properties of documents*, i.e., it consists of labels attached to resources. The Dublin Core [Weibel 1999], for example, is a 15-element dictionary that is used to catalogue resources in digital libraries. Related to this effort, the Warwick framework [Lagoze *et al.* 1996] emerged from a need for interoperability between different metadata sets. The Warwick framework thus provides a container architecture supporting the aggregation of different metadata sets, or packages. The Platform for Internet Content Selection (PICS) [Resnick and Miller 1996] supports the rating of on-line resources according to arbitrary scales. While PICS was first developed to prevent access to inappropriate material by children (without requiring centralized censorship), it is a general architecture that could be used for other purposes. Finally in the third case, the metadata can describe *relationships between documents*. For example, the Resource Description Format (RDF) [Lassila 1998] is an emerging architecture for representing metadata, which proposes a general graph-based data model for making assertions about Web resources. While XML offers a standard syntax for metadata, RDF proposes a standard way to express semantics. The two standards are however closely related, as XML is generally used to serialize RDF models (there are however other ways to represent RDF models, including a well-defined graphical notation). Historically, RDF evolved from another metadata architecture, the Meta Content Framework (MCF).

This article focuses on the last of these three usage patterns and on the idea of describing the structure of Web sites in a machine understandable way. An XML application designed for that purpose, the Structured Graph Format (SGF), is introduced. A framework developed around the format specification is also described. This framework encompasses the generation, the distribution and the processing of SGF metadata. Various issues related to these different aspects are discussed. One particular application of the SGF framework is then considered. It is explained how the availability of a formal description for the structure of a Web site makes it easier both to track and to represent the activity occurring within that site. For this reason, the SGF framework provides a foundation for building systems that support *social awareness* on the WWW.

The remaining sections of this article are organized as follows. Section 2 is dedicated to the SGF framework and to its components. The SGF format and its underlying data model are introduced. Its specification is then illustrated with a simple example. The framework integrates different categories of components, in particular SGF consumers and SGF producers that respectively process and generate the metadata. The generation of metadata is discussed first, and three approaches to that problem are compared. The role of SGF consumers is illustrated with SGViewer, a tool developed to support user navigation by dynamically generating interactive sitemaps. Section 3 introduces the notion of social awareness on the WWW, and argues for the need of better ways of grasping on-line activity. An overview of the related literature in Computer Supported Cooperative Work is proposed. It is then explained how in order to enable awareness on the Web, it is necessary (i) to track on-line activity, and (ii) to provide users with representations of this activity. The discussion shows that the availability of SGF metadata offers significant benefits for addressing these two issues. First, because SGF supports the definition of *regions* within a Web site, at different granularities, it makes activity tracking much more scalable. Second, because SGF metadata enables the graphical representation of Web sites, it thus provides an efficient way to represent the activity occurring within these sites.

2. Describing the structure of Web sites with the SGF framework

As mentioned before, a current effort is made to augment the WWW with metadata, with the aim of supporting automatic processing of on-line resources. Three different usage patterns for metadata are the description of: (i) the content of resources, (ii) the properties of resources, or (iii) the relationships between resources. This section focuses on the third situation and proposes a method for describing the structure of Web sites in a machine understandable way. The Structured Graph Format (SGF) is an XML application that was designed for this purpose. Moreover, a framework [Liechti *et al.* 1999a] was developed around the SGF format specification. This extensible collection of software components supports the generation, the distribution and the processing of SGF metadata.

An overview of the framework, in figure 1, shows that the different components belong to four categories. First, the *SGF format*, specified by the *SGF Document Type Definition (DTD)*, provides a way of describing the structure of Web spaces in *SGF documents*. Second, *SGF consumers* are software tools that read and analyze SGF documents for some purpose. The framework does not specify what this purpose should be. However, some SGF consumers have already been implemented and give an idea of possible applications. Third, *SGF producers* are software tools that generate SGF documents describing the structure of some Web space. While the framework does not specify how the producers should generate the metadata, several tools based on different approaches have already been developed and

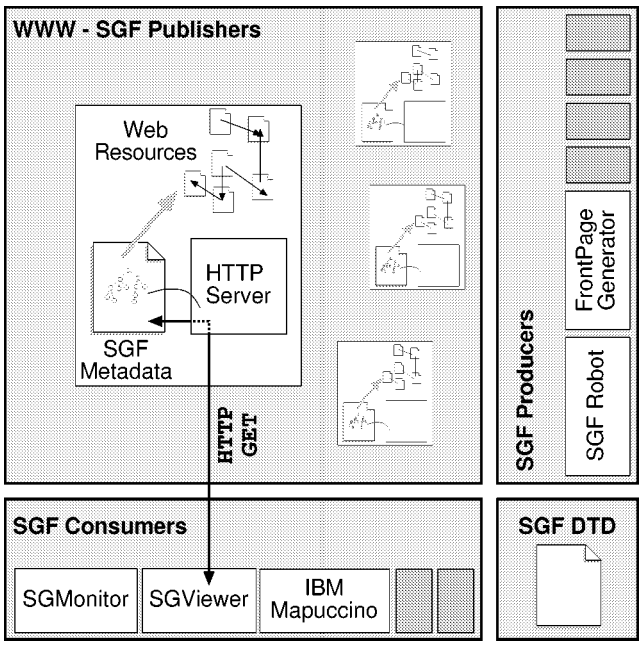


Figure 1. The SGF framework.

are available. Finally, *SGF publishers* are organizations and individuals who make SGF metadata available on their Web site. According to these roles, the following processes are supported by the framework:

- SGF publishers use SGF producers to create a formal description of the structure of their site. This description is encoded in an SGF document; it is thus a resource that can be published via a standard HTTP server.
- SGF consumers send HTTP requests to the servers maintained by SGF publishers, in order to fetch SGF documents. When they receive these documents, they analyze and process them for their own purpose. An SGF consumer can of course gather documents provided by several publishers.

In the following paragraphs, the SGF format is introduced and its specification is illustrated with a simple example. Three approaches to the generation of SGF metadata are then compared and a number of SGF producers are described. Finally, one application that benefits from SGF metadata is illustrated with SGViewer, a tool that supports user navigation with interactive site maps.

2.1. The Structured Graph Format

The central component in the framework is the Structured Graph Format (SGF), a metadata markup language that was first introduced in [Liechti *et al.* 1998]. SGF is an XML application that can be used to describe the structure of a Web site. The format gets its name from its underlying data model, which is a structured graph [Sifer and Potter 1996]. Structured graphs were initially developed in the context of project planning and later applied to software engineering and multimedia data modeling [Lowe *et al.* 1996]. They were designed to support the scalable browsing and editing of very large graphs. This explains our motivation to apply them to the WWW, which indeed is a very large network, often difficult to browse and manipulate. When complex information is organized in one or more hierarchies, it is easier to explore and understand [Botafogo *et al.* 1992; Mukherjea *et al.* 1995]. Structured graphs provide a way to organize a network in one or several overlapping hierarchies, making this network easier to manipulate. Applying structured graphs to the WWW thus means organizing an hypertext network into one or several hierarchies. SGF is a language that can be used to encode a structured graph, in other words a network and its overlapping hierarchies. Simply stated, a structured graph is defined by three sets: (i) a set of nodes, (ii) a set of hierarchical links between the nodes, and (iii) a set of associative links between the nodes.

Describing the structure of a site in SGF translates into transforming a directed graph into a structured graph. The source graph is defined by a set of nodes (the documents in the site) and a set of links (the hyperlinks between these documents). In the resulting structured graph, the set of nodes remains the same, but the set of links is split into two disjoint subsets of *hierarchical* and *associative* links. The classification of hyperlinks between these two categories is achieved by observing the following rule: when a hyperlink captures an *aggregation* relationship between two pages, it is marked as hierarchical. In the other case, it is marked as associative. This classification is generally easy to make, as most Web sites have an implicit hierarchical structure [Durand and Kahn 1998; Mizuuchi *et al.* 1999]. The home page is generally the root of a tree, and each page contains a number of links allowing visitors to move down the hierarchy. For example, somebody visiting an academic site will first access the university home page, then move down to a particular department, then down to a research group, etc. This tree is very often augmented with associative links. In the previous example, such links could exist between different departments or between different group members.

The classification of hyperlinks into hierarchical and associated links leads to one hierarchy, which denotes the structure of the hypertext (i.e., the relationships between the nodes). In other words, for each hierarchical link, there is a corresponding hyperlink. However, there are other ways to impose hierarchies on a Web site, by defining *hierarchical classifications* based on documents content and properties. For instance, one can think of classifications by author, creation date, popularity, topic, etc. Because it is based on structured graphs, SGF does support the definition of multiple overlapping hierarchies over a graph. An advantage of providing multiple hierarchies is that they offer multiple perspectives over information [Botafogo *et al.* 1992; Mukherjea *et al.* 1995]. They also support simple Boolean queries: as illustrated in figure 2, one can for example be interested in all documents created by X (using an author classification) during a time frame T (using a creation date

classification). These documents are those which appear at the same time under X in the author classification and under T in the time classification. If an appropriate technique is used to visualize an SGF document, the user might be able to formulate and answer these kinds of queries more easily. This is the case with SGViewer, a tool that will be described later. This article, however, will not discuss the use of multiple hierarchies in SGF further.

As it is the case with any XML-based markup language, the specification of the Structured Graph Format is given in the form of a Document Type Definition (DTD). The SGF DTD is given in table 1 and illustrated with a simple example in figure 3. An SGF document (<STRUCTUREDGRAPH>) consists of three parts: (i) a set of nodes (<NODES>), (ii) a set of hierarchical links (<HIERARCHY>) and (iii) a set of associative links (<NETWORK>). The nodes are defined by <NODE> elements, which specify a unique ID. The links are defined by <LINK> elements, which specify the unique IDs of the source and destination nodes. Besides, <SGATT> attributes are used to specify application specific properties, for example the URL of documents. An extended discussion of the SGF DTD can be found in [Liechti *et al.* 1998].

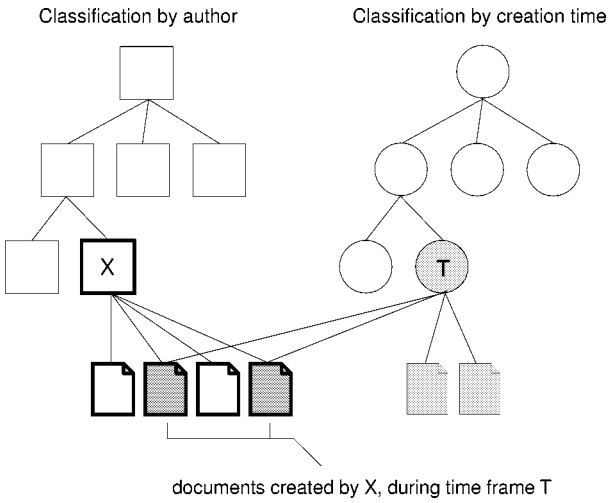
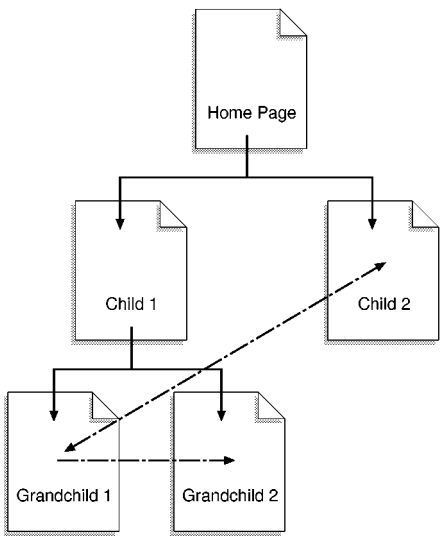


Figure 2. Using overlapping hierarchies to answer simple queries.

Table 1
The SGF Document Type Definition.

```
<!-- SGF DTD -->
<!-- an 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, with 2 attributes:  id and label -->
<!ELEMENT NODE (SGATT*) >
<!ATTLIST NODE
 NODEID ID #REQUIRED
 LABEL CDATA #IMPLIED >

<!-- a link between 2 nodes, with 3 attributes:
     source, destination and label -->
<!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 >
```




```
<STRUCTUREDGRAPH>
  <NODES>
    <NODE NODEID="N1" LABEL="Root">
      <SGATT NAME="URL" VALUE="http://..."/>
    </NODE>
    <NODE NODEID="N2" LABEL="Child 1">
      <SGATT NAME="URL" VALUE="http://..."/>
    </NODE>
    <NODE NODEID="N3" LABEL="Child 2">
      <SGATT NAME="URL" VALUE="http://..."/>
    </NODE>
    <NODE NODEID="N4" LABEL="Grandchild 1">
      <SGATT NAME="URL" VALUE="http://..."/>
    </NODE>
    <NODE NODEID="N5" LABEL="Grandchild 2">
      <SGATT NAME="URL" VALUE="http://..."/>
    </NODE>
  </NODES>
  <HIERARCHY>
    <LINK SOURCE="N1" DEST="N2"></LINK>
    <LINK SOURCE="N1" DEST="N3"></LINK>
    <LINK SOURCE="N2" DEST="N4"></LINK>
    <LINK SOURCE="N2" DEST="N5"></LINK>
  </HIERARCHY>
  <NETWORK>
    <LINK SOURCE="N4" DEST="N5"></LINK>
    <LINK SOURCE="N4" DEST="N3"></LINK>
    <LINK SOURCE="N3" DEST="N4"></LINK>
  </NETWORK>
</STRUCTUREDGRAPH>
```

Figure 3. A simple SGF example.

One of the benefits of SGF is that it supports the definition of *regions* within a Web site via selection operations. These operations permit selecting groups of nodes using both the hierarchical and associative relationships that exist between them. They include operations for selecting the *descendants* and the *ancestors* of a node in the hierarchy, and operations of selecting the *inputs* and the *outputs* of a node in the network. In each case, the scope of the selection can be specified by indicating a number of *levels*. For a given node, it is possible for example to select all its descendants, or only the two first levels of its descendants (children and grandchildren), or any arbitrary number of levels of its descendants. With these selection operations, it is possible to define subsets of related documents in the site and to manipulate them as single entities. It is also possible to define new regions by applying union, intersection or difference set operations to existing regions and nodes. In the academic site example, it is for example possible to define the "department" region as the union of the department home page and of all its descendants.

2.2. Generating the metadata with SGF producers

The widespread use of metadata on the WWW has much promise. Indeed, when looking at any metadata based architecture or system, it is generally easy to agree with its benefits. Unfortunately, this is not enough to guarantee its success and general adoption. The problem is that metadata has a considerable cost, which has generally to be supported by information publishers. There are different approaches to reducing this cost. In some cases, it is possible to automatically generate the metadata. This requires either that the information already exists in some form that can be interpreted by a software tool (possibly in a relational DBMS), or that the information is inferred by proper heuristics. The automatic propagation of classification metadata, using the connective structure of the Web, has been proposed in [Marchiori 1998]. When automatic generation is not possible, then special tools and editors should be provided to make the authoring process as easy as possible [Murray-Rust 1997; Waugh 1998].

Three methods for generating SGF metadata, illustrated in figure 4, are now introduced and compared. The tradeoff between the quality of metadata and the effort needed to produce it is highlighted. Manual authoring and automatic inference of the site structure are two of these methods. Yet, we argue for the benefits of the third one, where the metadata generation is smoothly integrated with the site management process. This ensures high quality metadata with less effort.

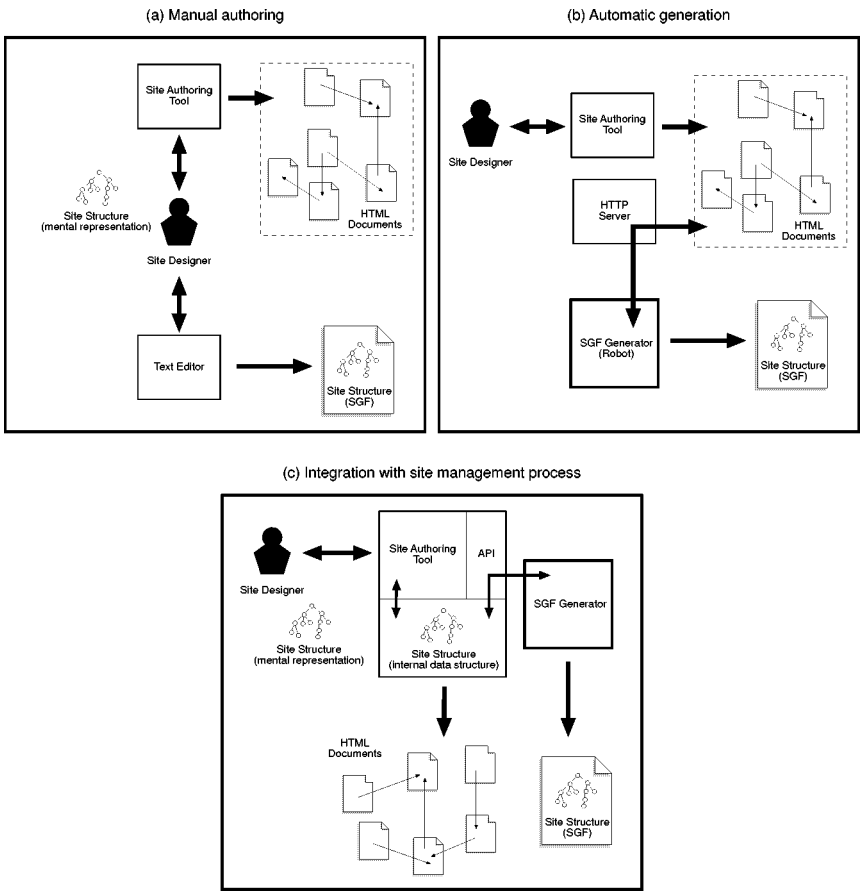


Figure 4. Three methods for generating SGF metadata.

2.2.1. Manual authoring

According to the W3C, metadata on the WWW should be machine understandable *and* human readable. This means that metadata should be encoded in a form that can be easily interpreted by humans, hence in text documents. This is the case for all XML applications and for SGF in particular. For this reason, it is possible to use a simple text editor to create SGF documents. The site publisher can thus manually create a description of the site structure. The advantage of this approach, illustrated in figure 4(a), is that the SGF description very precisely reflects the structure that the site publisher has in mind. The reason is that s/he is explicitly making the classification between hierarchical and associative links, clearly identifying aggregation relationships between documents. The major drawback of this solution is that it does not scale *at all*. It is only viable for very small sites, or portions of larger sites that do not change often. Otherwise, the generation and the maintenance of metadata are too burdensome. To resume, manual authoring theoretically leads to high quality metadata but requires too much effort (and thus practically leads to low quality metadata). Our own experiences have shown that it very quickly becomes an impractical solution.

2.2.2. Automatic generation: generic and site-specific heuristics

Instead of manually encoding the structure of the site, it is a better idea to develop an automatic solution. In this case, the site publisher designs and maintains the site as usual and periodically performs an operation to generate a new SGF document describing the site structure (figure 4(b)). This can be achieved by launching a software robot (also known as crawler or wanderer) that will visit the site by recursively following internal hyperlinks. On the basis of some rules, or heuristics, the robot will then infer the nature of hyperlinks, distinguishing hierarchical from associative links. Two approaches can be used to achieve this goal: the first one consists of proposing a *generic* solution, the second one consists of developing a *site-specific* solution.

A generic solution is one that can be used to infer the structure of *any* Web site. Its great advantage is that it requires very little effort from site publishers. They simply have to get an off-the-shelf tool and run it to become SGF publishers. The problem is that because of the many differences between Web sites, it is quite difficult to come up with a solution that will provide good results in all cases. The situation is made worse by the fact that many Web sites contain erroneous HTML documents, or simply have a poor organization. Despite that fact, general heuristics for extracting hierarchies in Web sites [Durand and Kahn 1998; Mizuuchi *et al.* 1999] and hypertexts [Botafogo *et al.* 1992] have been proposed and used with some success. We also have implemented different robots for automatically generating SGF documents. A common strategy is to perform a breadth-first traversal of the site starting from the home page. For every hyperlink it finds in the parsed HTML documents, it checks whether the target URL has already been visited. If this is not the case, the link is marked as hierarchical, otherwise it is marked as associative. This simple algorithm gives relatively good results and takes advantage of the implicit hierarchical organization of many sites. An approach used by another SGF robot consists in analyzing the file path in the documents URLs. In large sites, publishers generally use folders to organize their documents in the hierarchical file system. This hierarchical organization can be reconstructed by analyzing the URLs and can sometimes be used to classify the hyperlinks. To summarize, the advantage of generic and automatic generation is that it does not require much effort from site publishers. However, the results are not necessarily always very good and might not exactly reflect the structure that the publisher has in mind. The publisher can however modify the metadata generated by the robot to improve its quality.

The other solution for the fully automatic generation of SGF metadata is to develop a site-specific solution. In this case, the site publisher knows how the site is organized and is able to define better heuristics. Web programming languages such as WebL [Kistler and Marais 1998] and crawler development toolkits such as SPHINX [Miller and Bharat 1998] can be used to implement site-specific agents with less effort. Moreover, the site publisher might also use information stored in local databases to improve the quality of the metadata. Compared to the previous solution, this one requires some initial effort from the publisher (defining and implementing the heuristics). But when this initial effort is made, periodically generating a description of the site is easy. Besides, the quality of the metadata is likely to be higher.

2.2.3. Integration with site management process

Two methods for generating SGF metadata have been described. They exhibit a tradeoff between the cost and the quality of metadata. While manual authoring requires too much effort, automatic inference of the site structure does not always produce optimal results. For these reasons, we argue for the benefits of a third solution, where the generation of metadata is smoothly integrated with the site authoring process. Over the last few years, many Web authoring tools have appeared on the market. Some of these tools allow users to easily produce individual HTML documents. Others support the management of an entire site. This is the case of products such as Microsoft FrontPage[1] and NetObjects Fusion[2]. An interesting feature of these tools is that they allow users to define a hierarchical navigation structure, used to automatically generate a collection of navigation hyperlinks on the pages. It is a good idea to use this information, which is specified at design time, to generate the SGF description of the site. In this case, there is a nice balance between the quality of the metadata and the effort required from the site designer. Hence, the SGF

[1] URL: `http://www.microsoft.com/frontpage`.
[2] URL: `http://www.netobjects.com`.

description closely corresponds to the structure that the site designer has in mind, because s/he explicitly defines it. At the same time, because the metadata generation process is transparently integrated with the authoring process, no extra effort is required. Because the two previously mentioned authoring tools propose an API that gives access to their internal data structures, it is possible to implement SGF producers based on this approach (figure 4(c)). We have implemented a tool that communicates with Microsoft FrontPage to retrieve the navigation structure defined by the site designer, and translates it in SGF metadata. In our experiences, this approach to the generation of SGF metadata has proven to be the most efficient. It ensures high quality at low cost. Its main limitation is that it requires that the site designer uses an authoring environment that can generate SGF. It also means that the entire site needs to be managed with this single authoring environment.

2.3. Processing the metadata with SGF consumers

The motivation for defining the SGF format, and then developing the SGF framework, was the assumption that an application independent description of Web site structures could be useful on different occasions. While it is not exhaustive, the following list gives a few examples of applications that would benefit from such metadata:

- *Visualizing Web spaces.* There are different reasons for creating a graphical representation of a Web space. The first is to support user navigation, as discussed in the following paragraph. Mapuccino [Hersovici *et al.* 1998] is a dynamic site mapping system developed at IBM Research that is able to read SGF metadata [Ben-Shaul *et al.* 1999]. More generally, the visualization of a site provides an interface to better understand both its *structure* (i.e., the relationships between documents) and the *properties of its documents*. For example, a diagram can indicate what documents are accessed the most, what documents have not been updated for a long time, etc. In some cases the visualization can also provide an interface to apply operations on the site documents.
- *Applying operations on document collections.* We have discussed how it was possible to define regions within an SGF structure. Once defined, these regions can be manipulated as single entities. Instead of having to successively apply the same operation on several documents, it is then possible to select a region and to apply the operation only once. For example, editing operations can allow to copy, move or delete regions in the site. Another example of operation would consist of applying a formatting style to a region. In that case, the designer would first create a collection of style sheets, before specifying which one of them should be used for each region in the site.
- *Monitoring activity in Web spaces.* The WWW has become a social space where a lot of activity is taking place. Maintaining awareness about this activity seems important, but is unfortunately quite difficult. Supporting awareness makes it necessary firstly to capture some form of activity, and secondly to provide a representation of this activity. There are two ways in which SGF can make this process easier. First, it is generally not practical to capture the activity occurring within a whole site, nor the activity occurring within a single document. The notion of region supported by SGF thus provides a scalable and flexible way to capture activity at an intermediate level. Second, the graphical representation of a site that can be generated on the basis of SGF metadata can be annotated to represent the captured activity. These topics will be discussed in more details in section 3.

Although they may have very different intents, all SGF consumers share a common functionality. First, they must be able to fetch SGF metadata provided by SGF publishers. This means that they have to act as HTTP clients, sending requests to HTTP servers and processing their answers. Fortunately, the development of such tools has become very easy thanks to programming languages such as Java and to network programming libraries widely available. Second, these tools must be able to parse SGF documents and to verify their correctness. One of the benefits of XML being a standard syntax for all metadata formats is the availability of many tools supporting this function. There are in fact two kinds of XML parsers: non-validating and validating parsers. Non-validating parsers only verify that documents conform to the XML specification, validating parsers also verify that documents conform to a particular DTD (e.g., that they are proper SGF documents). A number of XML parsers are available, sometimes even freely, and can thus be used to facilitate the development of SGF consumers. A first application for SGF metadata is considered in the next section. First, the problem of navigation in hypertext is introduced. Then, SGViewer is described as an SGF consumer that uses the metadata to generate interactive site maps.

2.3.1. Creating visual representations of the site structure to support user navigation

Navigation problems in hypertext systems have been studied for a long time [Conklin 1987; Edwards and L 1989; Mukherjea and Hara 1997; Nielsen 1990]. When users explore an information space by following hyperlinks, they very often have problems maintaining a context and easily get *lost in hyperspace*. A partial solution to this problem consists in providing a 2D or 3D overview diagram of the hypertext structure, often referred to as a site map. Such a diagram can help the user by making it easier to understand where s/he is, has been and wants to go. A list of requirements for useful site maps, as well as a taxonomy for existing visualization techniques of hypertext structures has been proposed in [Durand and Kahn 1998]. A classification of visualization techniques for cyberspaces[3] is

[3] URL: `http://www.cybergeography.com`.

also made available by Martin Dodge, at the University College of London. This cyber-geography atlas includes a section dedicated to site maps. Because Web sites are generally organized hierarchically, a number of tools have been developed to represent hierarchical site maps, including WebToc [Nation *et al.* 1997], Nif-T-Nav [Jones 1996] and MAPA [Durand and Kahn 1998]. Moreover, visualizing the structure of a Web site relates to the general problem of visualizing large hierarchical and graph structures. Several approaches have tried to solve this problem by combining an overview of the structure with a smaller, but more detailed portion of it. These techniques are often referred to as focus+context; they include fisheye lenses [Furnas 1986], hyperbolic maps [Lamping *et al.* 1995] and perspective walls [Mackinlay *et al.* 1991].

In order to provide visitors with a site map, a common method consists of embedding a textual or graphical representation of the site into an HTML document. Such a static map is then rendered by the browser. Another approach, which we are arguing for, consists in (i) only providing an abstract description of the site structure (i.e., metadata) and (ii) using client-side visualization tools in conjunction with the browser. These tools can be either stand-alone applications or Java applets. Their role is to request the SGF metadata provided by the visited sites and to dynamically render it. Putting the emphasis on an abstract representation of the site structure presents a number of advantages:

- *Easier maintenance*. When the structure of a site changes, the map representing it has to be modified. When the site map is manually created by the site publisher, as it is the case for many clickable image maps, the maintenance can be very time consuming. In this case, there are two risks. First, the publisher can decide that the map will only give an overview of the site, which is often static and will therefore not need much maintenance. However such a map is not necessarily very useful, because it does not provide enough detail. Second, the publisher can decide to create a detailed map. But if maintenance is too much time consuming, the publisher will not always be able to keep up with it. This would result in a useless map, which does not accurately represent the structure of the site. On the other hand, maintaining an abstract description of the site is less demanding, as it can be automated more easily.
- *User-specific map vs. site-specific map*. To exaggerate the current situation on the WWW, every time a user visits a new site, s/he finds a different kind of map. This is a problem, as it requires repeated effort from the user to learn how to interpret these maps. A better approach is to let the user choose one visualization technique, supported by a specific tool, and to use this technique to display the map of *any* visited site. This of course requires that the visited sites provide an abstract description of their structure, in a format understood by the visualization tool. It is therefore important to agree on a standard representation for the structure of Web sites that could be used by such tools.
- *Interactive maps and client-side exploration*. Most sites provide either a graphical or a textual map, which in either case is static. The idea of client-side tools fetching structural metadata fosters the use of interactive visualization techniques. This results in maps that support a wide range of functions (e.g., summarization, filtering, searching) and that can be used as a new way to explore the site. After downloading the structure description, the user can look for useful information only by interacting with the map and without having to request intermediate documents to the server. This is particularly advantageous on slow connections, when following a hyperlink requires more than a few seconds.

2.3.2. SGViewer: an SGF consumer that dynamically generates interactive site maps

The first SGF consumer, named SGViewer [Liechti *et al.* 1998], was developed with the goal of making navigation in Web sites easier. When proposing SGF, our goal was to provide an application independent way to describe the structure of a site. Consequently, most of the existing techniques for representing hierarchies and networks can be adapted to visualize SGF documents. Our visualization tool, SGViewer, however uses an original focus+context technique. SGViewer visually represents the hierarchical and associative relationships specified in SGF metadata in two distinct panes. The first pane shows the site hierarchy, while the second shows a local network of associative links. Animated transitions occur when the user, starting from a global overview of the structure, successively zooms in and out to get more detailed information. Multiple synchronized windows can also be opened at the same time and thus give different perspectives of the structure to the user (see figure 7). Snapshots of SGViewer are given in figures 5–8; they show that:

- The upper frame in the window represents the hierarchical organization of the site, where each rectangle represents a document. The root of the site is drawn at the top of the frame. When a *hierarchical* link exists between say document A and document B, then document B is drawn *below* document A. In figure 5, a hierarchical link can for example be identified between the “SGF Home Page” and the “Publications” nodes.
- The lower frame in the window represents a local view of the site network. Again, each rectangle represents a document. Whenever an *associative* link connects document A to document B, then document B is drawn on the *right* side of document A. An associative link can be observed in figure 5, between the “FAQ” and the “SGViewer” nodes. The document represented at the center of the network can be changed by selecting a node in the hierarchy. In figure 5, the SGViewer node has been selected in the hierarchy and thus appears at the center of the network in the lower frame.

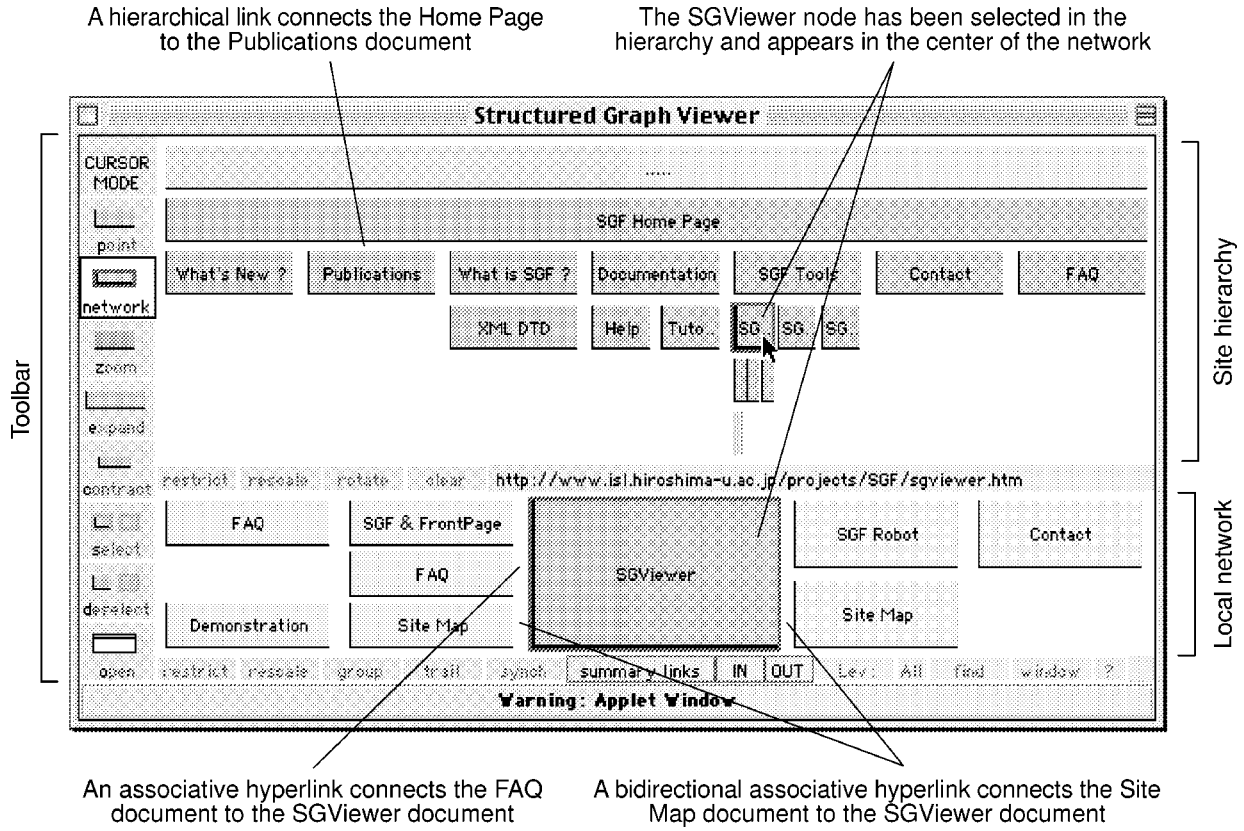


Figure 5. Visualizing SGF metadata with SGViewer.

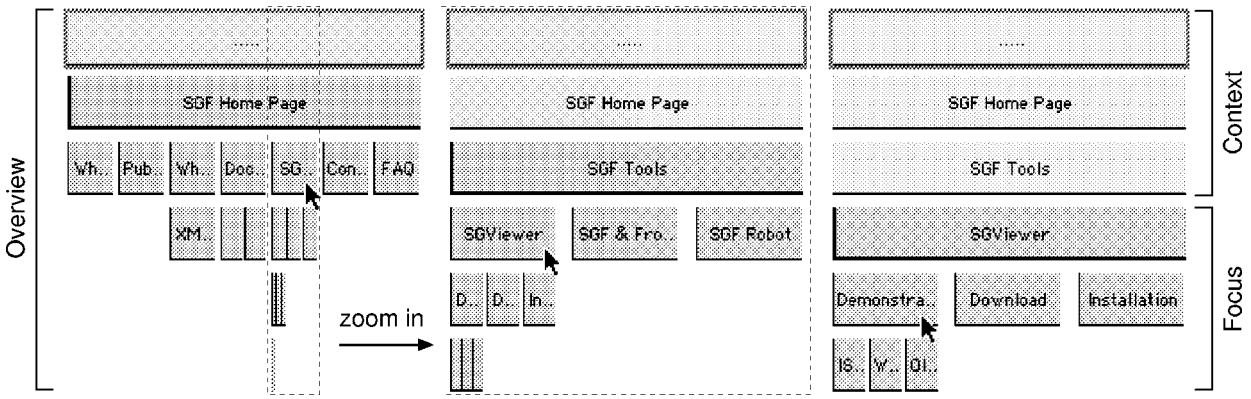


Figure 6. Zooming down the site hierarchy to get detailed information while maintaining some context.

- The toolbar on the left side of the window is used to perform different operations on the structure. For example, the user can zoom in and out the hierarchy, as shown in figure 6. It is also possible to expand and shrink different portions of the structure. The user can also select portions of the structure and restrict the diagram to only show these, as illustrated in figure 8. The search for the occurrence of keywords within the site is also supported. Finally, the tool, when running as an applet, communicates with the browser. This allows the user to fetch a URL in the browser by clicking on the corresponding node in the site map.

As explained in section 3, SGViewer can be used as a foundation for other SGF consumers. The maps that it generates simply represent the structure of the site, but they can be modified and annotated in different ways to represent other information (e.g., the current traffic on the site). Further, the tool provides an interface for selecting regions within the site and for applying operations on them (see section 3.2.2 and figure 9). The source code of SGViewer is available under an open source license on the SGF Web site[4].

[4] URL: `http://www.isl.hiroshima-u.ac.jp/projects/SGF/`.

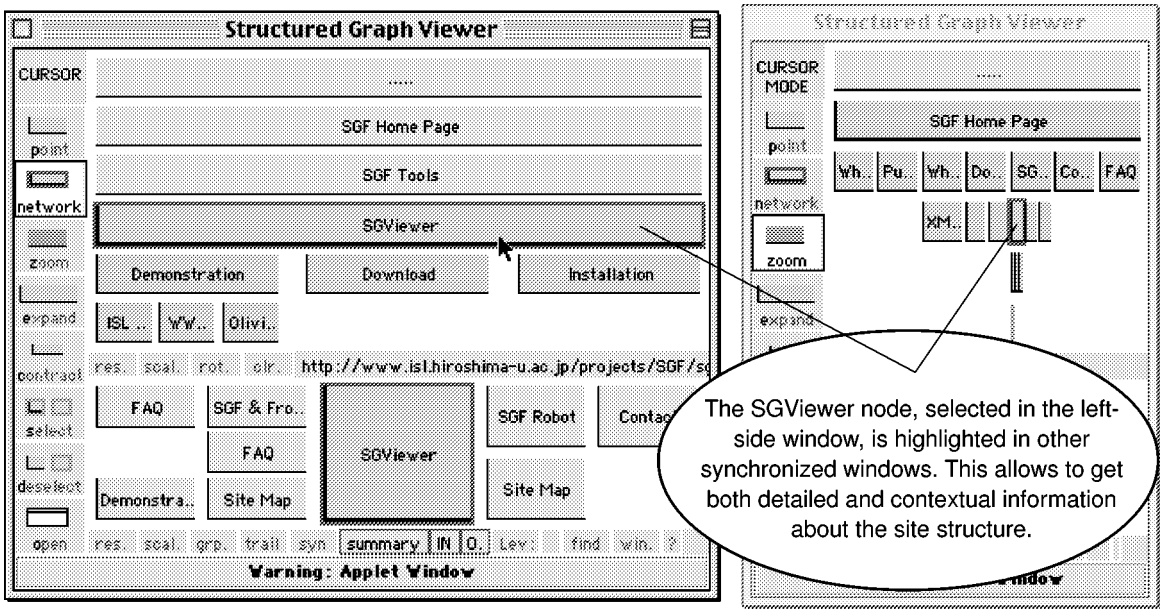


Figure 7. Using multiple synchronized windows.

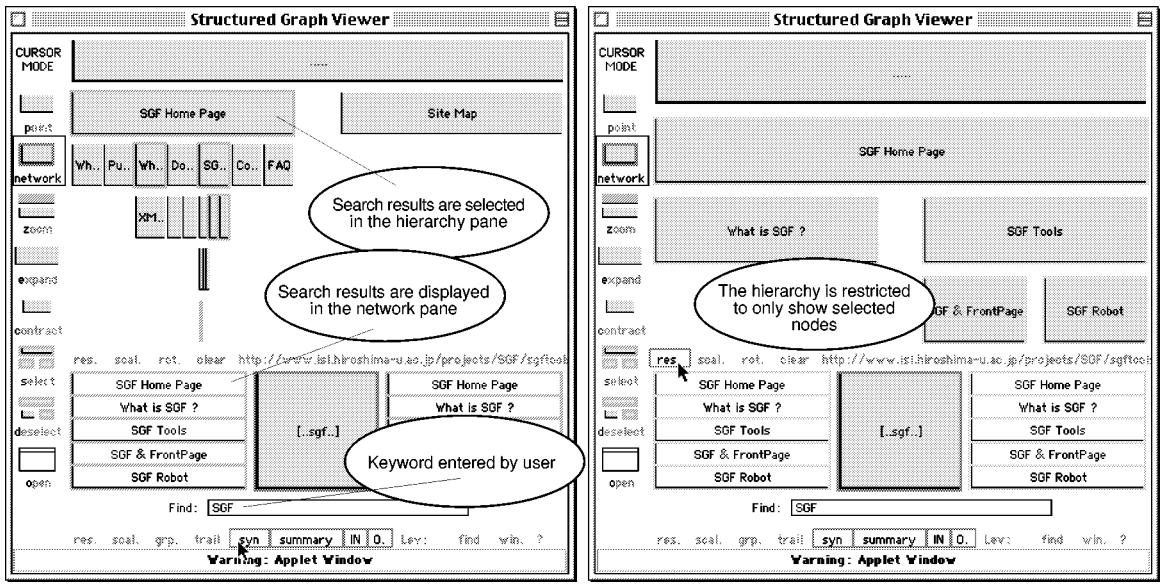


Figure 8. Restricting the hierarchy to display only selected nodes.

3. Using SGF to support social awareness on the WWW

The previous section has given an overview of the SGF framework, and explained how it encompasses the generation, distribution and processing of metadata describing the structure of Web spaces. It was suggested that the framework could support a range of different applications. Supporting user navigation with interactive site maps was described as one of these applications. This section focuses on another application that benefits from SGF metadata. The discussion shows that the framework provides a foundation for building systems that enable *social awareness* on the WWW. In other words, it explains how a formal description for the structure of a Web site makes it easier to grasp and understand the activity that is occurring within that site. Awareness is first introduced as a multifaceted notion extensively studied in the CSCW literature, which can be applied to the WWW in different ways. It is then explained how in order to enable awareness on the Web, it is necessary (i) to capture on-line activity, and (ii) to provide users with a representation of this activity. In both cases, the availability of SGF metadata offers significant benefits. First, it supports the definition of regions within a Web space, at different granularities, and thus ensures the scalability of monitoring systems. Second, it enables the graphical representation of Web spaces, which can be annotated in different ways to represent the captured activity.

3.1. Awareness in CSCW systems and on the WWW

The WWW was originally designed to support distributed groups and is thus, in essence, a collaborative technology. It is also a unique platform that offers many advantages when it comes to implementing CSCW systems [Dix 1996]: ubiquitous access, platform independence, reduced development costs, etc. A number of collaborative systems have thus been implemented on top of the Web infrastructure [Bentley *et al.* 1997; Girgensohn *et al.* 1996; Tollmar *et al.* 1996], sometimes augmenting the document-based interaction with synchronous communication facilities [Trevor *et al.* 1997; Walther 1996]. Also, WebDAV [Fielding *et al.* 1998; Whitehead and Wiggins 1998] is a new standard for distributed authoring and versioning on the Web. Further, many of the theoretical and empirical results reported in the CSCW literature can be applied to WWW technologies and Web-based systems. This is particularly true for the notion of *awareness*, defined by Dourish and Bellotti as "*an understanding of the activities of others, which provides a context for your own activity*". Awareness has been extensively studied and identified as a key feature for collaborative systems. Unfortunately, it is generally not supported on the Web and we have previously argued for the need to address this issue [Liechti *et al.* 1999b].

The first motivation for increasing social awareness on the WWW follows the observation that the very nature of this system has considerably evolved over the last decade. At first, the system was *a huge document repository*, giving access to only static information. Following a period of very rapid growth, unpredicted application domains such as electronic commerce and distance education soon emerged. With them appeared the need for interactivity, and a number of extensions were made to the original models and protocols. At this point, the Web became *an integrated collection of services*. Furthermore, the Web is now evolving towards a *social space* [Donath and Robertson 1994], where people do not conceptually access documents, but where they visit *places* and meet each other. Hence, social activity is increasingly occurring on the Web, and it seems important to be able to better grasp and understand this activity.

In virtual shops, for example, salespeople should be made aware of the presence of customers and should be able to interact with them. In virtual schools, students should be made aware of the presence and availability of their teachers. Teachers should also be able to monitor the activity of their students, in order to assist them more effectively. More generally, many people have now acquired a *territory* in cyberspace, if only by publishing a personal homepage. Although it still requires a substantial effort, they should be interested to know what is happening in this territory. Who is visiting my site? How long are they staying? What do they think? Are they coming back? These are all interesting questions that any Web publisher should be able to easily answer. Indeed, various tools and techniques supporting some of these functions already exist. Simple examples include page hit counters, guest books and HTTP access logs statistical analyzers. Nevertheless, there seems to be a need for richer, more interactive and synchronous ways to monitor activity on the Web. There also seems to be a need for more natural and seamless means to bridge the Web with the physical environment: as we are able to *sense* people in our physical neighborhood, we should also be able to *sense* people in our digital neighborhood.

As a matter of fact, the notion of awareness is a very broad one, and a number of CSCW systems have exhibited various categories of awareness. These can be applied to the WWW in very different ways. *Workspace awareness* [Gutwin and Greenberg 1998; Gutwin *et al.* 1996], for example, is supported by some collaborative editors and tools. The idea is to help users understand how the other participants are interacting in a shared workspace, by using proper user interface techniques and widgets (e.g., telepointers and radar views). Because browsing the Web can be a collaborative activity, special browsers, such as MetaWeb [Trevor *et al.* 1997] and GroupWeb [Greenberg and Roseman 1996] have been developed. They allow their users to know what other participants are doing and what they are looking at. They also allow a participant to control the browsers of other users in order to guide them. As discussed in [Kobayashi *et al.* 1998], providing customer support on the Web is one application for collaborative browsing.

Another kind of awareness is referred to as *group awareness*. Here, the idea is to give coworkers a general idea of each other's status and activity. Coworkers are very often interested to know where their peers are, if they are busy or not, etc. They use this information to decide when and how it is appropriate to interact with them. NYNEX Portholes [Girgensohn *et al.* 1996] and @Work [Tollmar *et al.* 1996] are two systems that use the WWW as an implementation platform for enabling group awareness in distributed teams. Digital cameras periodically take a snapshot of each member's workspace. These snapshots are gathered, assembled and augmented with some information (e.g., daily schedule, short messages, etc.). The resulting mosaic is encoded in HTML and published on a Web server. Thus, users have easy access to a panel at which they can occasionally glance to have a general idea of the group's current situation. Other systems, such as those described in [Palfreyman and Rodden 1996] and [Sidler *et al.* 1997], have tried to make "nearby" Web users aware of each other. Because people accessing the same pages, the same sites, or simply the same kind of information are likely to share common interests, it is probably a good idea to allow them to meet each other. This idea of facilitating unplanned encounters and impromptu interaction has been investigated in various CSCW systems, including Piazza [Isaacs *et al.* 1996], OfficeWalker [Obata and Sasaki 1998], and FreeWalk [Nakanishi *et al.* 1996].

Finally, an important notion is the one of *peripheral awareness*, which is related to the natural ability of peo-

ple for monitoring what is happening around them, even while they are focusing their attention on some task. Media spaces are CSCW systems that use sensors (cameras, microphones, etc.), displays (screens, speakers, etc.) and high-speed networking to create a continuous media link between distant locations. Although media spaces were first developed to support explicit communication, their evaluation suggests that their major benefit is probably a support for peripheral awareness [Bly *et al.* 1993]. Using media spaces, people are able to constantly and effortlessly have an idea of what their peers are doing. Peripheral awareness also plays an important role for what Weiser and Brown [1996] call calm technology: "*Calm technology engages both the center and the periphery of our attention, and in fact moves back and forth between the two.*" It is also central in the AROMA media space [Pedersen and Sokoler 1997], which provides users with an abstract representation of the activity at the remote site. The work of Ishii and Ullmer [1997] on ambient media is also closely related to this idea. We have also developed several systems for monitoring activity on the Web that put an emphasis on peripheral awareness [Liechti and Ichikawa 1999; Liechti *et al.* 1999b]. Our goal in these projects was to create a seamless bridge between the physical and computational worlds, by providing different abstract representations of the activity captured on the Web.

Implementing a system for supporting awareness can be decomposed into three main tasks. The first task consists of capturing various events that indicate some form of activity. The second task consists of analyzing and integrating these events to build and maintain the state of a model. Finally, the third task consists of providing one or several representations of this model, and eventually in taking actions following particular changes in the state of this model. This process can be applied in very different settings, both on the WWW and in the real world. For example, consider the members of a team who constantly want to know the current physical location of each other. The first task can be achieved by implementing a user location tracking system, for example with active badges sensors [Want *et al.* 1992]. The second task consists of gathering the data collected by the sensors in order to maintain a simple model, i.e., a relation between the set of workers and the set of locations. Finally, the third task consists of providing representations for this model. One can easily imagine a graphical representation of the physical environment (a map), with moving objects indicating the current location of each person. This tool could also raise an alert (e.g., by playing a sound) each time the model state changes in a way which indicates that a person has just arrived. As a matter of fact, this simple real world scenario can be easily transposed to the WWW. This requires first to track the activity of people visiting a site, and second to represent this activity. Problems related to these two tasks are examined next, and the role of SGF metadata in addressing these problems is then described.

3.2. Capturing activity on the WWW

In order to grasp high-level behaviors on the Web, for example to determine that a good customer is coming back, or that most visitors have been unhappy with a particular section of the site, it is necessary to capture and to integrate low-level events. These low-level events ultimately result from the interaction between a user and a Web browser. This interaction consists of actions that fit into two categories: (i) actions that involve GUI components embedded in a document (e.g., clicking on a button in an HTML form or interacting with a Java applet), and (ii) actions that indicate the user's change of location on the Web (i.e., jump to another URL). Different mechanisms are needed to monitor these two types of actions. For the first category, the monitoring is achieved directly in the code controlling the user interface embedded in a particular document (e.g., in a Java applet or client-side script). The actions are thus captured locally and have to be explicitly notified to the monitoring system. An example is an applet that displays a button and notifies the monitoring system each time this button is clicked by the user.

For the second category, monitoring is achieved by capturing and analyzing HTTP requests sent from clients to servers. This can be done either on the server side, on the client side, or somewhere in between if proxies are used. The concept of *intermediazies* [Barret and Maglio 1998], defined as computational elements that lie along the path of Web transactions, is an interesting approach to that problem. Generally, each HTTP request is associated with a user movement on the Web and the last request sent by a client indicates the current position of its user. Moreover, several HTTP requests can be integrated both *vertically* and *horizontally*. A *vertical* integration is achieved by constructing the list of URL's successively visited by a particular user, in other words the path followed by this user. This method has been used to improve the performance of dynamic content generation by Web servers [Schechter *et al.* 1998]. It has also been used to support collaborative filtering [Chalmers *et al.* 1998]. A *horizontal* integration is achieved by constructing the list of all users currently located in the same region of the WWW. In this case, it is possible to indicate to the users that there are other people around and eventually to offer them a way to interact. This method has been used in systems such as [Palfreyman and Rodden 1996; Sidler *et al.* 1997]. We now review a number of methods that can be used to capture HTTP requests.

3.2.1. Capturing HTTP requests to monitor activity on the WWW

The first solution for capturing requests on the server side is obvious: it simply consists in implementing a new HTTP server, or in modifying the source code of an existing one. In this case, the developer can of course detect and process any request sent by clients. While it is very flexible, this method has evident shortcomings, as it requires a substantial implementation effort. Besides, adding support

for awareness should be possible on top of any infrastructure, and should not be limited by the choice of a particular server. Still on the server side, the second solution consists of using the access logs generated by most of the HTTP servers. These are text files, where each line contains the description of a request processed by the server. This description includes the client IP address, the requested URL, the time, the result code returned by the server, and some other information. This method is appropriate for generating access statistics and is used by a variety of tools. The problem is that it is not always appropriate for synchronous monitoring. Also, systems administrators are sometimes reluctant to give access to these log files.

The third solution for capturing requests consists of using the HTTP proxy mechanism. In this case, the browser is configured not to address requests directly to HTTP servers, but to an intermediate server called the proxy. The proxy receives requests from many clients, forwards the requests to the target servers on the WWW and subsequently forwards their answers to the clients. This approach is interesting, because it allows to track the activity of a community of users very easily, without having to change the sites visited by this community. Imagine a workgroup, with their browsers configured to send their requests to the same proxy server. The software running on this proxy receives HTTP requests from all group members, and is thus able to know what site each person has visited. This raises a number of privacy issues, but also offers benefits for the users. One can for example think of a collaborative filtering system that automatically builds a list of the most interesting sites for the group by observing each member's behavior.

The fourth solution consists of using server-side scripting (e.g., CGI or Java servlet), or a combination of server-side and client-side scripting (e.g., JavaScript). The Common Gateway Interface (CGI), proposed to support dynamic generation of information on the Web, has been extensively ued for a long time. Every CGI is associated with a URL and is triggered on the server in response to HTTP requests targeting this URL. The script has access to a number of parameters characterizing the triggering request (client IP, time, etc.). A particularly interesting parameter is the HTTP referrer, which indicates the URL from which the request was sent. Depending on the situation, this information can be interpreted in two different ways. In the first situation, there is an <HREF> hyperlink targeting the CGI in a URL "A". When a user clicks on the link, the request is sent to the server, the CGI is executed and the resulting HTML stream is displayed in the browser. In this case, the value of the HTTP referrer parameter is the URL "A" (i.e., the document that the user was reading just before). In the second case, an HTML document "B" contains an `<IMG>` tag, which specifies as the image source URL not a graphical file (e.g., GIF or JPEG), but a CGI script. In this case, when the browser fetches the document and renders it, it first sends a request to the server to retrieve the image, which thus triggers the CGI. At this point, the value for the HTTP referrer is the URL "B" (i.e., the document that the user is reading now). This mechanism allows one to track users location by inserting pseudo-images (i.e. IMG tags that trigger a CGI) in the HTML documents of a site. The CGI script will be triggered by each user access, will thus be able to retrieve the HTTP referrer parameter and to notify other system components.

3.2.2. Ensuring scalability of awareness systems with SGF regions

The goal of the awareness systems that we are developing is to help users gain a better understanding of what is happening on their Web site. In these systems, we generally use the CGI mechanism because it does not require the modification of the HTTP server software. The drawback is that it requires a modification of the HTML documents to include the pseudo-image tag that triggers the CGI. However, this is not really an issue for the sites we monitor, as they make use of templates and Server-Side Includes (SSI). By placing the pseudo-image tag in the header file that is automatically included by every document, we only have to change one HTML file to enable user tracking.

Although it is relatively easy to detect HTTP requests, and so to track the movements of users on the site, scalability issues quickly appear. When the size of the site grows beyond a certain limit, and particularly when the traffic is relatively heavy, it is not meaningful to consider the site as a single entity. In other words, it is not sufficient to know that "somebody is accessing my site now". Instead, it is necessary to have some idea of *where* in the site this person is. On the other hand, tracking activity on a single document is not very meaningful either. For example, it is generally more interesting to know that "there are 100 people in the research section of my site", rather than "there are 2 people on this URL, 3 people on this one, 1 person on this one, etc.". What is needed to address this issue of scalability is a mechanism for defining groups of related documents, at different granularities. Using this mechanism, users can then define different regions within the site. They can then track the activity occurring within these regions independently, in a very flexible way.

One possibility for defining such regions is to explicitly build lists of URL's. But this is obviously a tedious task, which moreover has to be repeated each time new documents are added to the site. However, as we explained in section 2.1, the SGF metadata format supports the definition of regions via *selection operations*. With these operations, it is possible to select groups of nodes using both the *hierarchical* and the *associative* relationships that are defined among them. It is possible to select the *descendants* and the *ancestors* of a node in the hierarchy, as well as the *inputs* and the *outputs* of a node in the network. Besides, if a graphical tool is used to display an SGF document, the regions can be defined interactively. SGViewer, for example, supports the definition of regions with the interface illustrated in figure 9. In this particular case, the user has defined one region (Important pages) as the set of three groups of nodes. While two of these groups contain

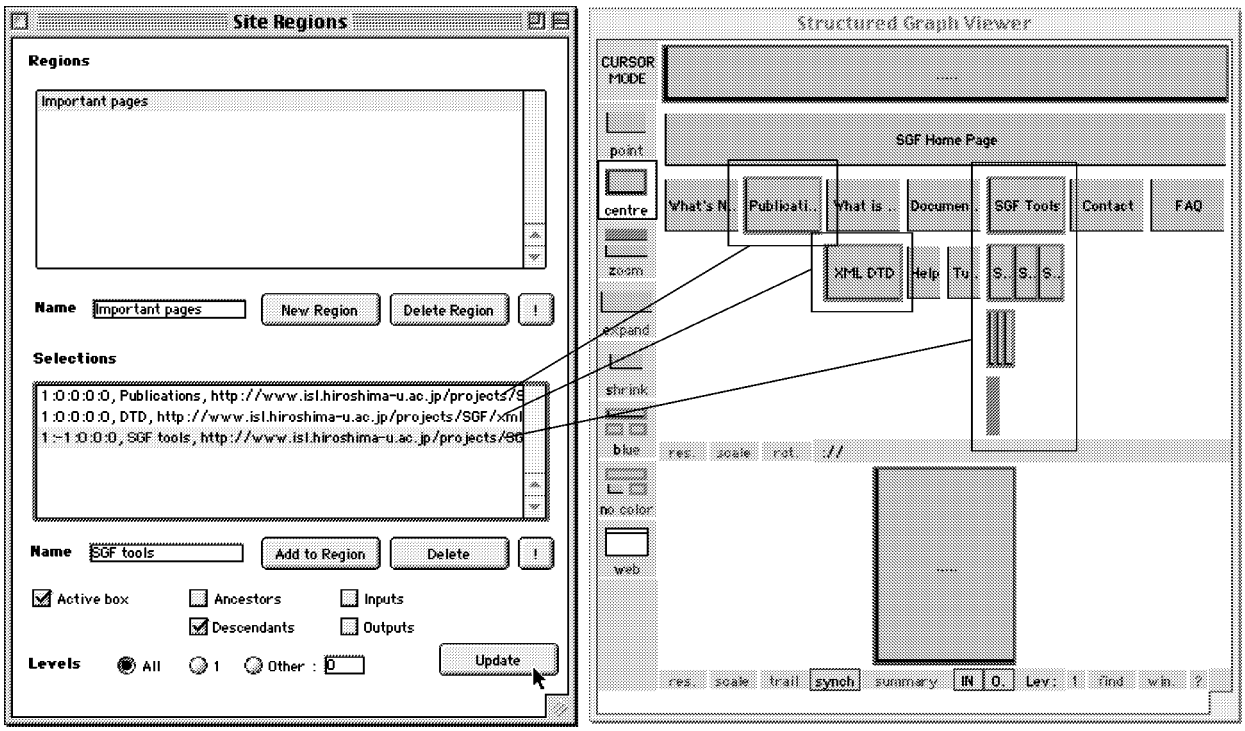


Figure 9. Interactive definition of regions with SGViewer.

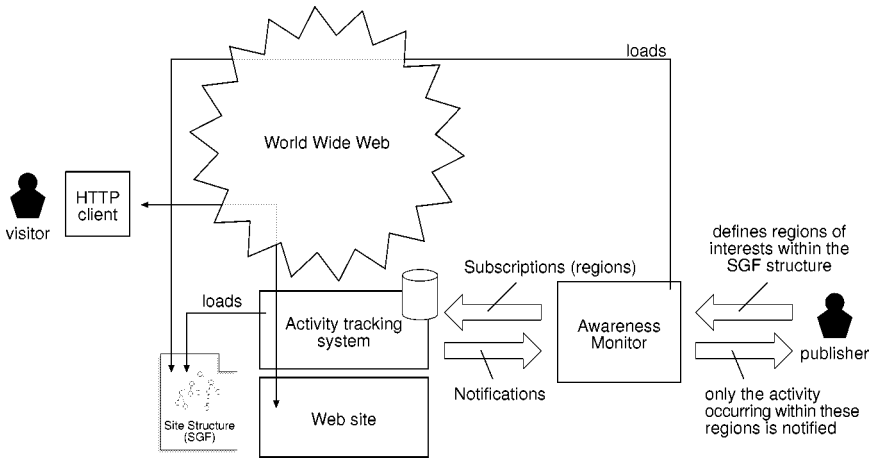


Figure 10. Architecture of an awareness system using SGF metadata.

a single document (DTD and Publications), the third one contains an entire sub-tree in the site hierarchy (the SGF Tools node and all its descendants).

Implementing awareness systems on Web sites that provide SGF metadata becomes much easier, because users have a way of specifying which portions of the site they are interested in. They can also monitor several regions at the same time, but in slightly different ways. For example, they might decide that every visitor in a region A will be notified with a low-pitched sound, while every visitor in a region B will be notified with a high-pitched sound. The architecture of an awareness system exploiting SGF metadata is given in figure 10. The site publisher interacts with an awareness monitor, which has fetched the SGF description of the monitored site (it is thus an SGF consumer). With this tool, s/he defines regions of interest and issues subscriptions to the activity tracking system. When visitors are detected on the site, notifications are sent to the awareness monitor, but only if the user location is within one of the regions specified in the subscriptions. The event can then be notified to the publisher.

3.3. Representing activity captured on the WWW

The purpose of an awareness system is to let its users know that some activity is occurring. In the case of the WWW, this means that users should be provided with representations of the activity occurring within their site. These representations can be created in very different ways. Some

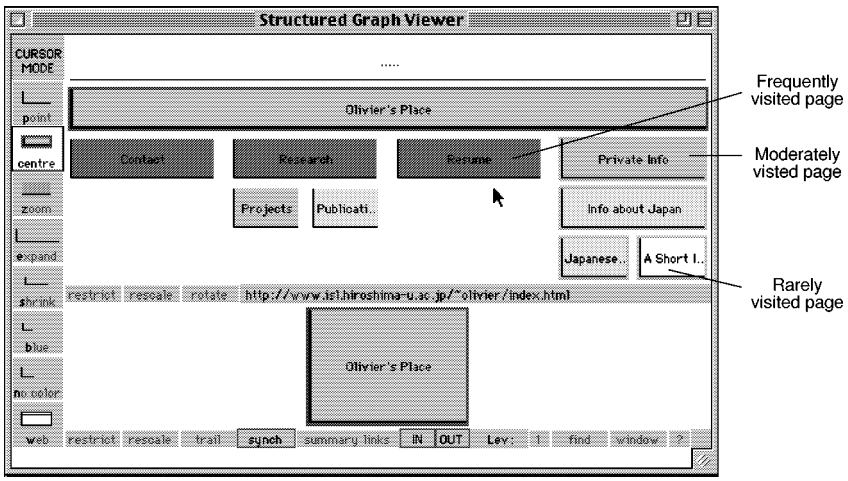


Figure 11. Representing current traffic with color attributes in SGMonitor.

of them can be very *abstract*, only giving a general idea of what is happening and necessitating some interpretation effort. Some can be very *explicit*, providing a detailed description of the captured events. In the former case, the user might for example be able to tell that "there are many people in the site". In the latter, they might be able to tell that "the user using this IP address is accessing this URL for the second time".

These two different approaches have their own benefits. In fact, they complement each other and are most effective when used in concert, at different stages of the same process. Abstract representations are particularly useful in the pre-attentive stage, when users are concentrating their attention on a task not related to monitoring (e.g. reading a paper at their desk). Abstractions make it easy to maintain a peripheral awareness of what is happening, without requiring much effort [Pedersen and Sokoler 1997]. When an event is perceived and judged critical by the user (often by a noticeable discontinuity in the abstract representation), s/he may bring the awareness tool at the center of his attention, and search for a explanation. At this point, an explicit representation will be more appropriate, because it will provide more detailed information. In other words, abstract representations are well suited to create a continuous and seamless link between the real and computational words. On the other hand, explicit representations are more appropriate when it comes to *analyzing* what has happened. We thus believe that awareness tools should combine both abstract and explicit representations of Web activity, allowing the user to move the monitoring activity between the center and the periphery of his/her attention.

We have implemented several systems that use abstract representations of Web activity. In the first system [Liechti *et al.* 1999b], the user interacts with a window virtually opened on his Web site. When the window is opened, the activity of users is notified by a variety of natural sounds. Different sound features (e.g., pitch, volume, duration) are used to represent different kinds of events. For example, the visitors in one region of the site might be notified with birds chirps, while the visitors in a different one might be notified with dog barking. In the second system [Liechti and Ichikawa 1999], we proposed the idea of using dynamic photographs (compositions of photo-realistic components) to represent the captured activity. In one example, a variable number of people is overlaid onto a background scene to map the traffic on the site. By watching the virtual crowd, it is possible to have some idea of the current activity. In another example, the photograph of a tree is used to represent both the traffic and feedback received from visitors. While the quantity of traffic affects the size of the tree (heavy traffic will make the tree grow), the nature of the feedback affects the color of its leaves (green leaves indicate positive comments, red leaves indicate negative ones).

In order to create explicit representations of Web activity, a common solution is to use standard diagrams (bar charts, pie charts, etc.). This approach is used by a number of tools, which process the logs generated by HTTP servers to generate statistics. While such representations are useful, they often make it difficult to situate the information in context. However, we have explained that the availability of SGF metadata made it possible to create a visual representation of the site structure. When such a map is created, it can be annotated in various ways to represent ongoing activity. In this case, the information provided by the monitoring tool is easier to understand, as it is provided in the context of a site map. Users can thus more easily identify interesting regions, and compare how different portions of the site relate to each other.

SGMonitor is a slightly modified version of SGViewer, which was implemented to represent current traffic on a Web site. The tool is thus an SGF consumer, which first gets an SGF description of the monitored site. It is also connected to an activity tracking system and plays the role of an awareness monitor as in figure 10. When visitors are detected on the site, notifications are sent to SGMonitor and the diagram is accordingly updated. As illustrated in figure 11, the color of the nodes is modified to indicate the current traffic: dark nodes are visited by many users, while light ones are only visited by a few. The color of nodes is only one graphical attribute that can be modified

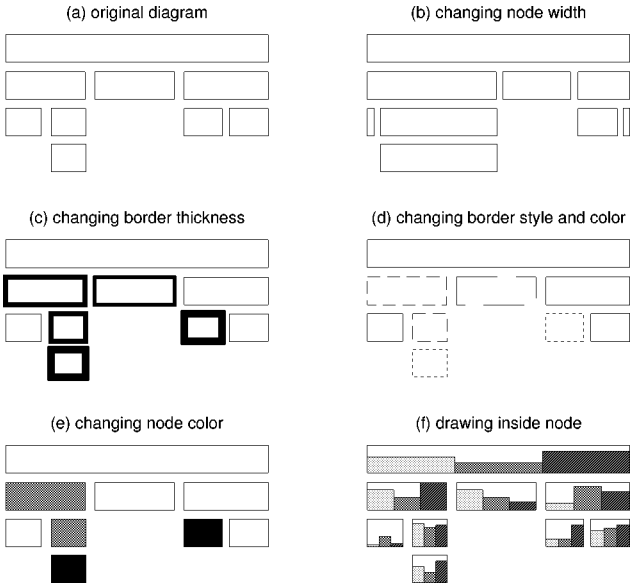


Figure 12. Representing activity with modified SGViewer diagrams.

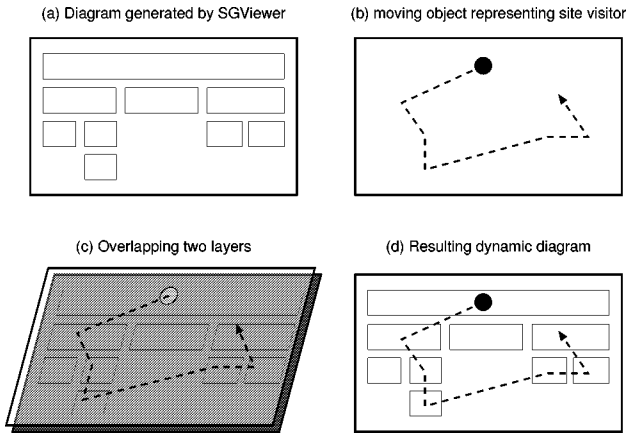


Figure 13. Representing activity in a dynamic layer overlapping the SGViewer diagram.

to annotate a diagram generated by SGViewer. Some others are described in figure 12, making use both of the nodes inside, border and width. Another possibility for indicating activity within SGViewer is to keep the diagram unchanged, but to overlap a dynamic layer on top of it. As shown in figure 13, visitors can for example be represented with moving circles. In this case, it is possible to follow a visitor as s/he explores the site.

4. Conclusion

The widespread use of metadata on the WWW might be one of the most important steps in its evolution. It will lead to what is now called the "Semantic Web", an information space that can be browsed by humans, but also by software agents. The automatic processing of Web resources enabled by metadata will benefit many applications, particularly in digital libraries and electronic commerce settings. In this article, we explained how metadata could be used to describe the structure of Web sites in a machine understandable way. The Structured Graph Format (SGF) was described as an XML application used to describe a Web site as a structured graph. Around this format specification, the SGF framework was designed as an extensible and integrated collection of software components that support the generation, distribution and processing of SGF metadata. Three approaches to the generation of metadata were compared, highlighting a tradeoff between quality and cost. SGViewer, a tool that generates interactive site maps on the basis of the metadata, was also described.

We then argued for the need to increase social awareness on the Web, in other words to make it easier to grasp the activity which is occurring on-line. The notion of awareness was introduced and situated in the context of the Computer Supported Cooperative Work. It was explained how this multi-faceted notion can be applied to the WWW in different ways. We then showed that the SGF framework provides a foundation and offers significant advantages for building awareness systems. First, the formal description of a Web site structure increases the scalability of monitoring systems. By supporting the definition of regions within a site, at different granularities, it allows the users to monitor activity in very flexible ways. Second, the site maps generated on the basis of the metadata provide an efficient way to represent activity on the site. These explicit representations of activity were contrasted with abstract representations. Explicit representations are useful for analyzing the activity that has been captured on the site. On the other hand, abstract representations provide an efficient way to seamlessly bridge the WWW with the physical environment. We argued that awareness tools should combine these two kinds of representations, allowing users to move the monitoring activity between the center and the periphery of their attention.

Acknowledgements

The first author was supported with a Japanese Government Monbusho Scholarship and the second author with a fellowship from the Japan Society for Promotion of Science. The authors would like to thank Jun Noritake who contributed to the project by implementing some of the SGF producers (robots) described in section 2.2.2.

References

Barret, R. and P.P. Maglio (1998), "Intermediaries: New Places for Producing and Manipulating Web Content," *Computer Networks and ISDN Systems 30*, 11–21, 509–518.

Ben-Shaul, I., M. Hersovici, M. Jacovi, Y.S. Maarek, D. Pelleg, M. Shtalhaim, V. Soroka, and S. Ur (1999), "Adding Support for Dynamic and Focused Search with Fetuccino," In *Proceedings of the WWW'8 International Conference*, Toronto, Canada.

Bentley, R., T. Horstmann, and J. Trevor (1997), "The World Wide Web as Enabling Technology for CSCW: The Case of BSCW," *Computer-Supported Cooperative Work: The Journal of Computer-Supported Cooperative Work 6*, 2–3, 111–134.

Bly, S., S. Harrison, and S. Irwin (1993), "Media Space: Bringing People Together in a Video, Audio and Computing Environment," *Communications of the ACM 36*, 1, 28–47.

Bosak, J. (1997), "XML, Java and the future of the Web," *World Wide Web Journal 2*, 4, 219–227.

Botafogo, R.A., E. Rivlin, and B. Shneiderman (1992), "Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics," *ACM Transactions on Information Systems 10*, 2, 142–180.

Chalmers, M., K. Rodden, and D. Brodbeck (1998), "The Order of Things: Activity-Centered Information Access," *Computer Network and ISDN Systems 30*, 1–7, 359–367.

Conklin, J. (1987), "Hypertext: An Introduction and Survey," *IEEE Computer 20*, 9, 17–41.

Dix, A. (1996), "Challenges and Perspectives for Cooperative Work on the Web," In *Proceedings of the ERCIM Workshop on CSCW and the Web*, Sankt Augusting, Germany.

Donath, J.S. and N. Robertson (1994), "The Sociable Web," In *Proceedings of the 2nd International WWW Conference*, Chicago, IL.

Durand, D. and P. Kahn (1998), "MAPA: A System for Inducing and Visualizing Hierarchy in Websites," In *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia*, pp. 66–71.

Edwards, D.W. and H. L (1989), *Lost in Hyperspace: Cognitive Mapping and Navigation in a Hypertext Environment*, Intellect Books, Oxford.

Fielding, R.T., E.J. Whitehead, Jr., K.M. Anderson, G.A. Bolcer, P. Oreizy, and R.N. Taylor (1998), "Web-Based Development of Complex Information Products," *Communications of the ACM 41*, 8, 84–92.

Furnas, G. (1986), "Generalized Fisheye Views," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '86)*, ACM Press, Boston, MA, pp. 16–23.

Girgensohn, A., A. Lee, and K. Schueter (1996), "Experiences in Developing Collaborative Applications Using the World Wide Web "Shell"," In *Proceedings of the 7th ACM Conference on Hypertext*, ACM Press, Washington, DC, pp. 246–255.

Greenberg, S. and M. Roseman (1996), "GroupWeb: A WWW Browser as Real Time Groupware," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '96)*, ACM Press, pp. 271–272.

Gutwin, C. and S. Greenberg (1998), "Design for Individuals, Design for Groups: Tradeoffs Between Power and Workspace Awareness," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '98)*, ACM Press, Seattle, WA, pp. 207–216.

Gutwin, C., M. Roseman, and S. Greenberg (1996), "A Usability Study of Awareness Widgets in a Shared Workspace Groupware System," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '96)*, ACM Press, Boston, MA, pp. 258–267.

Hersovici, M., M. Jacovi, Y.S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur (1998), "The Shark-Search Algorithm. An Application: Tailored Web Site Mapping," *Computer Networks and ISDN Systems 30*, 1–7, 317–326.

Isaacs, E.A., J.C. Tang, and T. Morris (1996), "Piazza: A Desktop Environment Supporting Impromptu and Planned Interactions," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '96)*, ACM Press, Cambridge, MA, pp. 315–324.

Ishii, H. and B. Ullmer (1997), "Tangible Bits: Towards Seamless Interfaces Between People, Bits and Atoms," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*, ACM Press, Atlanta, GA, pp. 234–241.

Jones, K.L. (1996), "NIF-T-NAV: A Hiearchical Navigator for WWW Pages," In *Proceedings of the 5th International WWW Conference*, Paris, France.

Kistler, T. and H. Marais (1998), "WebL – A Programming Language for the Web", *Computer Network and ISDN Systems 30*, 1–7, 259–270.

Kobayashi, M., M. Shinozaki, T. Sakairi, M. Touma, S. Daijavad, and C. Wolf (1998), "Collaborative Customer Services Using Synchronous Web Browser Sharing," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '98)*, ACM, Seattle, WA, pp. 99–108.

Lagoze, C., C.A. Lynch, and R.J. Daniel (1996), "The Warwick Framework: A Container Architecture for Aggregating Sets of Metadata," Technical Report TR96-1593, Department of Computer Science, Cornell University, Ithaka, NY.

Lamping, J., R. Rao, and P. Pirolli (1995), "A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, ACM Press, Denver, CO, pp. 401–408.

Lassila, O. (1998), "Web Metadata: A Matter of Semantics," *IEEE Internet Computing 2*, 4, 30–37.

Liechti, O. and T. Ichikawa (1999), "A Visual Interaction Mechanism for Increasing Awareness on the WWW," In *Proceedings of the IEEE International Symposium on Visual Languages*, IEEE Computer Society, Tokyo, Japan.

Liechti, O., M. Sifer, and T. Ichikawa (1998), "Structured Graph Format: XML Metadata for Describing Web Site Structure," *Computer Networks and ISDN Systems 30*, 1–7, 11–32.

Liechti, O., M. Sifer, and T. Ichikawa (1999a), "A Metadata Based Framework for Extracting and Using Web Sites Structures," In *Proceedings of the International Conference on Multimedia Computing and Systems (ICMCS '99)*, IEEE Computer Society, Florence, Italy, pp. 51–56.

Liechti, O., M. Sifer, and T. Ichikawa (1999b), "A Non-Obtrusive User Interface for Increasing Social Awareness on the World Wide Web," *Personal Technologies 3*, 3, 22–32.

Lowe, D.B., A. Ginige, M. Sifer, and J. Potter (1996), "The Matilda Data Model and Its Implications," In *Proceedings of the 3rd International Conference on Multimedia Modeling*, Toulouse, France.

Mackinlay, J.D., S. Card, and G. Robertson (1991), "Perspective Wall: Detail and Context Smoothly Integrated," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'91)*, ACM Press, New Orleans, LA, pp. 173–179.

Marchiori, M. (1998), "The Limits of Web Metadata and Beyond," *Computer Network and ISDN Systems 30*, 1–7, 1–9.

Miller, R.C. and K. Bharat (1998), "SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers," *Computer Network and ISDN Systems 30*, 1–7, 119–130.

Mizuuchi, Y. and K. Tajima (1999), "Finding Context Paths for Web Pages," In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, ACM Press, Darmstadt, Germany, pp. 13–22.

Mukherjea, S., J.D. Foley, and S. Hudson (1995), "Visualizing Complex Hypermedia Networks Through Multiple Hierarchies," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*, ACM Press, Denver, CO, pp. 331–337.

Mukherjea, S. and Y. Hara (1997), "Focus+Context Views of Word-Wide Web Nodes," In *Proceedings of the 8th ACM Conference on Hypertext and Hypermedia*, ACM Press, Southampton, UK, pp. 187–196.

Murray-Rust, P. (1997), "Jumbo: An Object-Based XML Browser," *World Wide Web Journal 2*, 4, 197–206.

Nakanishi, H., C. Yoshida, T. Nishimura, and T. Ishida (1996), "FreeWalk: Supporting Casual Meetings in a Network," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '96)*, ACM Press, Cambridge, MA, pp. 308–314.

Nation, D.A., C. Plaisant, G. Marchionini, and A. Komlodi (1997), "Visualizing Websites Using a Hierarchical Table of Contents Browser: WebTOC," In *Proceedings of the 3rd Conference on Human Factors and the Web*, Denver, CO.

Nielsen, J. (1990), "The Art of Navigating Through Hypertext," *Communications of the ACM 33*, 3, 296–310.

Nürnberg, P.J. and H. Ashman (1999), "What Was the Question? Reconciling Open Hypermedia and World Wide Web Research," In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, ACM Press, Darmstadt, Germany, pp. 83–90.

Obata, A. and K. Sasaki (1998), "OfficeWalker: A Virtual Visiting System Based on Proxemics," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '98)*, ACM, Seattle, WA, pp. 1–10.

Palfreyman, K. and T. Rodden (1996), "A Protocol for User Awareness on the World Wide Web," In *Proceedings of the ACM Conference*

on Computer Supported Cooperative Work (CSCW '96), ACM Press, Cambridge, MA, pp. 130–139.

Pedersen, E.R. and T. Sokoler (1997), "AROMA: Abstract Representation of Presence Supporting Mutual Awareness," In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*, ACM Press, Atlanta, GA, pp. 51–58.

Rajaraman, A. and P. Norvig (1998), "Virtual Database Technology: Transforming the Internet into a Database," *IEEE Internet Computing 2*, 4, 55–58.

Resnick, P. and J. Miller (1996), "PICS: Internet Access Controls Without Censorship," *Communications of the ACM 39*, 10, 87–93.

Schechter, S., M. Krishnan, and M.D. Smith (1998), "Using Path Profiles to Predict HTTP Requests," *Computer Networks and ISDN Systems 30*, 1–7, 457–467.

Sidler, G., S. A., and W. H. (1997), "Collaborative Browsing in the World Wide Web," In *Proceedings of the 8th Joint European Networking Conference*, Edinburgh.

Sifer, M. and J. Potter (1996), "Structured Graphs: A Visual Formalism for Scalable Graph Based CASE Tools," *Australian Computer Journal 28*, 1, 13–26.

Tollmar, K., O. Sandor, and A. Schoemer (1996), "Supporting Social Awareness @Work: Design and Experience," In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '96)*, ACM Press, Cambridge, MA, pp. 298–307.

Trevor, J., T. Koch, and G. Woetzel (1997), "MetaWeb: Bringing Synchronous Groupware to the World Wide Web," In *Proceedings of the ECSCW'97*, Kluwer Academic, Lancaster, UK, pp. 65–69.

W3C (1998a), MathML 1.0 Specification, `http://www.w3.org/TR/REC-MathML/.`

W3C (1998b), SMIL 1.0 Specification, `http://www.w3.org/TR/REC-smil/.`

Walther, M. (1996), "Supporting Development of Synchronous Collaboration Tools on the Web with GroCo," In *Proceedings of the ERCIM Workshop on CSCW and the Web*, Sankt Augusting, Germany.

Want, R., A. Hopper, V. Falcao, and J. Gibbons (1992), "The Active Badge Location System," *ACM Transactions on Information Systems 10*, 1, 91–102.

Waugh, A. (1998), "Specifying Metadata Standards for Metadata Tool Configuration," *Computer Networks and ISDN Systems 30*, 1–7, 23–32.

Webber, D.R. (1998), "Introducing XML/EDI Frameworks," In *EM – Electronic Transactions. EM – Electronic Markets*, Vol. 8, B.F. Schmid, D. Selz and R. Sing (Eds.) 1.

Weibel, S. (1999), "The State of the Dublin Core Metadata Initiative April 1999," *D-Lib Magazine 5*, 4.

Weiser, M. and J.S. Brown (July 1996), "Designing Calm Technology," *PowerGrid Journal Version 1.01*.

Whitehead, J. and M. Wiggins (1998), "WEBDAV: IETF Standard for Collaborative Authoring on the Web," *IEEE Internet Computing 2*, 5, 34–40.