Mandala: An Architecture for Using Images to Access and Organize Web Information

Jonathan I. Helfman

[1] AT&T Labs - Research, Shannon Laboratory
Human Computer Interaction Department
Room B255, 180 Park Avenue, Florham Park, NJ 07932-0971
[2] University of New Mexico, Computer Science Department
Room FEC 313, Albuquerque, NM 87131
jon@research.att.com
http://www.cs.unm.edu/~jon/

Abstract. Mandala is a system for using images to represent, access, and organize web information. Images from a web page represent the content of the page. Double-clicking on an image signals a web browser to display the associated page. People identify groups of images visually and share them with Mandala by dragging them between windows. Groups of image representations are stored as imagemaps, making it easy to save visual bookmarks, site indexes, and session histories. Image representations afford organizations that scale better than textual displays while revealing a wealth of additional information. People can easily group related images, identify relevant images, and use images as mnemonics. Hypermedia systems that use image representations seem less susceptible to classic hypertext problems. When image representations are derived from a proxy server cache, the resulting visualizations increase cache hit-rates, access to relevant resources, and resource sharing, while revealing the dynamic access patterns of a community.

1 Using Images to Represent Web Information

Most web software represents information textually. We use textual bookmarks and history lists in browsers, textual indexes and similarity metrics in search engines, and textual concept hierarchies in link taxonomies. Textual representations and organizations shape our experience of interacting with information[13].

On the web, a selectable image is a link to another resource. *Imagemaps* link to multiple resources. In general, a selectable image contains more data than a typical string of selectable text and has the potential to provide more of an indication if the link is relevant and worth following.

In many cases, images from a web page represent the content of the page. Even the images used for decoration, navigation, and advertisement often provide additional characterizations of a web site (although these can usually be recognized and suppressed). Web images may represent their context for many reasons. There is a rich history of using images to illustrate and illuminate

Fig. 1. Mandala client displaying three groups of representations.

manuscripts. Most technical documents contain highly descriptive illustrations, charts, or diagrams. On the web, image formats were standardized before audio or text stylesheets. As a result images are not only used for illustration, but they have become a common strategy for site differentiation.

Mandala lets people visualize large groups of web pages by displaying selectable thumbnails of the pages' images. Groups may be determined in many ways: the URLs in a bookmark file, the history of a browsing session, the results of a query, etc. Mandala's displays function as visual interactive indexes; they provide an overview of large amounts of information without sacrificing access.

Mandala automatically builds groups of representations and provides a user interface for viewing and editing groups of representations and saving them as imagemaps. A snapshot of a Mandala client is shown in Fig. 1.

Creating a system that uses images to represent web information requires solving several technical problems. While digital text enjoys an efficient representation, digital images require more memory and bandwidth, as well as support for compression, decompression, and scaling. Web-based systems also require support for HTTP monitoring and HTML parsing. Additional requirements include the ability for people to identify image groups visually and share them with the system easily. These technical challenges have been addressed by Mandala's modular architecture, which provides a flexible and general platform for visual information research on the web. Mandala's architecture allows it to function in multiple ways (e.g. as a GUI for organizing and maintaining information represented by images, a visual bookmark facility, an imagemap editor, a cache visualization tool, and a visual look-ahead cache).

Mandala has been fully implemented and is almost fully operational. Mandala's image server (see Sec. 2.1) is used to generate imagemaps for CoSpace, an experimental VRML system at AT&T Labs. Mandala's proxy server (see Sec. 2.2) is used to monitor web browsing sessions for PadPrints, a web navigation tool, which builds multi-scale maps of a user's browsing history[9].

The remainder of this paper describes Mandala's architecture, as well as some preliminary observations about using image representations.

2 Architecture and Implementation

Figure 2 illustrates Mandala's component structure. White boxes represent the Mandala components. Solid grey shapes represent standard components that are used without modification (e.g. web servers and browsers). Light grey boundaries indicate runtime constraints. Arrows indicate the exchange of HTTP-like messages consisting of ASCII headers and optional ASCII or binary data.

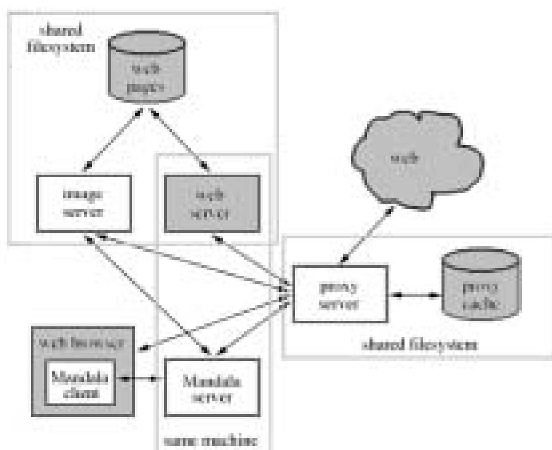

Fig. 2. Mandala's Component Structure.

Mandala is less concerned with supporting visual queries through feature extraction and indexing than previous Visual Information Systems[11]. Mandala's architecture does not preclude a component for image analysis and feature extraction, but web images are surrounded by a rich context of meta-information, which seems to provide an ample feature set for indexing[19].

2.1 Imago: Mandala's Image Server

Mandala's image server, called *Imago*, has been developed to support fast image data compression and decompression, image shrinking, imagemap creation, and image meta-data extraction (i.e. information about the image that is normally hidden in the image's header, such as its dimensions or total number of colors).

While the other Mandala components are written in Java, Imago is written in C, both for speed and reliability. Publicly available C code to read and write GIF and JPG images is relatively robust[3, 10] compared to the Java image decompression classes, which frequently throw undocumented exceptions when trying to decode GIF or JPG variations that they don't fully support.

Imago creates thumbnails and imagemaps according to client specifications. A minimal thumbnail specification is a URL for the input image, which causes Imago to use default scale factors and filtering functions. Clients can specify the maximum thumbnail dimension, filter function, return style (whether to return the URL or the data), and replacement style (whether to overwrite an exiting thumbnail of the same name or generate a new name). Additional options ignore input images that don't meet minimal requirements for size or number of colors.

Imago's default filter function uses a form of hierarchical discrete correlation (HDC) that scales the input by one half[5]. To scale images by arbitrary amounts, Imago uses successive invocations of HDC and a single invocation of more general image rescaling algorithm that creates and scales filter kernels based on the scale factor[20]. The possible slowness of the final pass is minimized because scale factors are guaranteed to be less than one half.

A minimal imagemap specification is a stub name for the output and a list of URL pairs for web images and associated resources. The imagemap command supports each of the thumbnail options described above, as well as layout style (grid, tile, random, user-set) and background style (solid color, tiled image).

Imago also rates imagemaps according to a heuristic that attempts to identify images that are used for decoration, navigation, or advertisement. The rating scheme is based on image meta-information as opposed to image content. It ranks large, square images with many colors higher than small, wide or tall images with few colors. The rate of an imagemap is the average rate of the imagemap's thumbnails. Ratings allow clients to identify imagemaps that are most likely to contain useful representations.

2.2 Mirage: Mandala's Proxy Server

Mandala's proxy server, called *Mirage*, has been developed to support local caching of images, transparent web browser monitoring, and HTML parsing. A proxy server is a program that sits between web servers and web clients, such as browsers (see [1] and [12] for good introductions to proxy servers). Requests are copied from clients to servers. Responses, if any, are copied back to clients and possibly cached, in case they are requested again. Proxy servers provide several benefits, such as reduced web latency and increased effective bandwidth[12], increased web access (when the destination server is down, but the resource is cached), and savings on long-distance connection charges[15].

The two main problems for a proxy server implementation are how to determine if a cached file is fresh and which files to remove when the cache gets full. Freshness is difficult to determine because few servers use the `Expires` HTTP

Fig. 3. Snapshot of a Dynamic Cache Visualization.

header and there is no way for a destination server to inform a proxy server when a resource has been updated [14]. When the `Expires` header is not used, Mirage uses a typical hueristic, which estimates freshness time as a factor of the last-modified time. If the server does not transmit a last-modified time, Mirage assigns the resource a maximum age. When the cache is full, Mirage uses a Least-Recently Used (LRU) algorithm for removing old files from the cache [18]. Researchers disagree about the performance of LRU [1, 17, 18], however LRU has a more efficient implementation than algorithms that need to compute scores and insert into sorted lists [8].

Mirage improves Mandala's performance by caching thumbnails and imagemaps generated by Imago and installed on a web server. Mandala also benefits from Mirage's extensions. Proxy server extensions are a recommended strategy for creating utilities and applications that enhance the experience of web browsing [2, 4]. Mirage has been extended to allow registered applications to monitor web browsing sessions[9]. Monitoring with a proxy server is transparent to users who need only configure their browser to use the proxy server via a command-line option or preference setting. Mirage has also been extended to describe its cache and parse HTML, capabilities that Mandala uses to identify references to links and images in the cached pages.

A snapshot of a cache visualization is shown in Fig. 3. Displaying selectable images from a proxy server cache has several surprising implications. Cache visualizations improve the cache *hit-rate* (i.e. the percentage of requests that can be satisfied out of the cache) more than algorithmic approaches, which rely on the chance that people will unknowingly request a page that someone else has requested[1]. Cache visualizations increase access to relevant information. When people can see the contents of the cache, and select an image to access an associated page, they are more likely to access information that is relevant to their needs or interests. When displaying images as they are cached, the cache visualizations reveal dynamic usage patterns of entire communities and promote unprecedented sharing of resources. Streams of similar images represent active hypertext trails of anonymous community members (e.g. the cars in Fig. 3). If any trail is of interest, selecting an image in the trail allows an observer to become a participant immediately, blazing a new trail, which is soon displayed as an additional stream of images in the communal montage.

2.3 Mandala Server

The Mandala Server automatically groups images based on origin (e.g. images from the same web server or from the same browsing session), but could be extended to use similarity of contextual information[19] or extracted features[11]. The Mandala Server also groups images from pages that are reachable from the page most recently requested by the user's web browser, a capability that allows a Mandala client to function as a visual look-ahead cache. Web searches initiated with any search engine define groups of images from pages that match the query.

The Mandala Server communicates with multiple Mandala clients by posting messages to a bulletin-board and sending the clients brief messages that they should check the bulletin-board when they have a chance. This protocol prevents the server from swamping the clients with a flood of messages and image data. Clients are free to read updates when user-activity decreases.

2.4 Mandala Client

Each Mandala client supports automatic layout and animation of image representations. User interactions include image selection, image positioning, and animation control. Clients may be full-fledged applications or Java applets (with slightly reduced functionality). Mandala clients communicate only with a single Mandala server. Double-clicking on an image causes the client to signal the Mandala server, which signals a web browser (via its client API) to display the associated page.

Mandala clients use separate windows for each group of representations (see Fig. 1). People edit imagemaps by repositioning thumbnails. People edit groups by dragging and dropping thumbnails between windows. People define new groups by dragging thumbnails into an empty window.

3 Preliminary Observations about Image Representations

With the increased access afforded by viewing large groups of image representations comes associated copyright and privacy concerns. Taking images off web pages may seem to be a copyright violation, but if they are used as an interactive index that promotes access to the information, then they are no more of an infringement than the textual indexes built by search engines. In both cases, the intent is to promote access to original information and encourage the livelihood of the purveyors of the original information. Privacy becomes a concern when people do not wish to share information about their browsing behavior with a community. In this case, they may choose not to use a proxy server, but then they will not benefit from the communal cache.

Image representations seem to diminish the effect of several classic hypertext problems. For example, *spatial disorientation* is caused by unfamiliarity with possibly complex hypertext structures[6], while *cognitive overhead* is caused by the general level of complexity associated with multiple choices[7]. Because image representations deemphasize hypertext structure, people navigate through clusters of similar images instead of structures of hypertext links. Images may provide better navigational cues than textually-labeled links because images have been shown to improve human memory for associated information[16].

Other hypertext problems include *lack of closure*, the inability to determine which pages have been visited or if any nearby unvisited pages are relevant, and *embedded digression*, the inability to manage multiple, nested digressions[7]. Because people have a remarkable memory for images, they can distinguish quickly between familiar and unfamiliar images[21]. A system that uses image representations may therefore help people identify new information when seeing an unfamiliar image and find previously accessed information by remembering and locating a familiar image.

The hypertext problem of *trail-blazing*, the inability to determine if a link is worth following, may be diminished when using image representations because images have the potential to provide a better indication if a link is worth following than brief textual labels. The hypertext problem of *session summarization*, the inability to save the state of a browsing session, is alleviated by storing the image representations associated with a browsing session as an imagemap, which provides a visual session summary that preserves the interactive nature of the browsing experience.

4 Conclusions

This paper describes Mandala, a system that provides visual interactive overviews of large amounts of information by using images from web pages to represent those pages (Sec. 1). Mandala's architecture is discussed and distinguished from earlier Visual Information Systems that support image indexing and visual querying (Sec. 2). The use of a proxy server for transparent monitoring of web browsers is described, and cache visualizations are reported to improve cache hit-rates and reveal dynamic communal access patterns (Sec. 2.2). In addition, image representations are shown to increase concerns over copyright and privacy while diminishing classic problems associated with hypertext (Sec. 3).

References

1. Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Stephen Williams, and Edward A. Fox. Caching proxies: Limitations and potentials. In *Proceed-*

ings of the Fourth International World Wide Web Conference, December 1995. http://www.w3.org/pub/Conferences/WWW4/Papers/155/ .
2. Rob Barrett and Paul Maglio. Intermediaries: New places for producing and manipulating web content. In *Proceedings of the Seventh International World Wide Web Conference*, 1998. http://wwwcssrv.almaden.ibm.com/wbi/www7/306.html.
3. T. Boutell. http://www.boutell.com/gd/.
4. Charles Brooks, Murray S. Mazer, Scott Meeks, and Jim Miller. Application-specific proxy servers as http stream transducers. In *Proceedings of the Fourth International World Wide Web Conference*, December 1995. http://www.w3.org/pub/Conferences/WWW4/Papers/56/.
5. Peter J. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16(1):20–51, 1981.
6. Jeff Conklin. A survey of hypertext. Technical Report STP-356-86, MCC, February 1987.
7. Carolyn Foss. Tools for reading and browsing hypertext. *Information Processing and Management*, 25(4):407–418, 1989.
8. Jonathan Helfman. Insights and surprises using image representations to access and organize web information. AT&T Web Implementor's Symposium, July 1998. http://www.cs.unm.edu/ jon/mandala/wis98/wis98.html.
9. Ron Hightower, Laura Ring, Jonathan Helfman, Ben Bederson, and Jim Hollan. Graphical multiscale web histories: A study of PadPrints. In *Hypertext '98 Proceedings*, pages 58–65, 1998.
10. Independent JPEG Group. ftp://ftp.uu.net/graphics/jpeg/.
11. Clement H. C. Leung and W. W. S. So. Characteristics and architectural components of visual information systems. In Clement Leung, editor, *Visual Information Systems, Lecture Notes in Computer Science 1306*. Springer, 1997.
12. Ari Loutonen and Kevin Altis. World-wide web proxies. In *Proceedings of the First International Conference on the World-Wide Web, WWW '94*, 1994. http://www1.cern.ch/PapersWWW94/luotonen.ps.
13. Marshall McLuhan. *The Gutenberg Galaxy: the Making of Typographic Man*. University of Toronto Press, 1962.
14. J. C. Mogul. Forcing http/1.1 proxies to revalidate responses, May 1997. http://www.es.net/pub/internet-drafts/draft-mogul-http-revalidate-01.txt.
15. Donald Neal. The harvest object cache in New Zealand. In *Proceedings of the Fifth International World Wide Web Conference*, 1996. http://www5conf.inria.fr/fich_html/papers/P46/Overview.html.
16. Allan Paivio. *Imagery and Verbal Processes*. Holt, Rinehart, & Winston, 1971.
17. Tomas Partl and Adam Dingle. A comparison of www caching algorithm efficiency. http://webcache.ms.mff.cuni.cz:8080/paper/paper.html.
18. James E. Pitkow and Margaret M Recker. A simple yet robust caching algorithm based on dynamic access patterns. In *Proceedings of the Second World Wide Web Conference*, 1994. http://www.ncsa.uiuc.edu/SGD/IT94/Proceedings/DDay/pitkow/caching.html.
19. Neil C. Rowe and Brian Frew. Finding photograph captions multimodally on the world wide web. In *AAAI-97 Spring Symposium Series, Intelligent Integration and Use of Text, Image, Video, and Audio Corpora*, pages 45–51, March 1997.
20. Dale Schumacher. General filtered image rescaling. In David Kirk, editor, *Graphics Gems III*, pages 8–16. Academic Press, 1994.
21. Lionel Standing, Jerry Conezio, and Ralph Norman Haber. Perception and memory for pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science*, 19(10):73–74, 1970.