ACM SIGOPS 5th European Workshop

NEEDED: A SYSTEMATIC STRUCTURING PARADIGM FOR DISTRIBUTED DATA

by Jerome H. Saltzer

Library 2000

M.I.T. Laboratory for Computer Science

June 25, 1992

The purpose of this note is to alert the distributed and operating systems communities to a research and design exercise that is going on in an adjacent application-oriented community. This research and design exercise involves inventing a paradigm for distributed system structuring that is distinctly different from that of remote procedure call and related paradigms that are the usual focus of the distributed systems community.

The class of applications involved might be roughly characterized as distributed, linked data. The remote procedure call paradigm, in which a client asks a server to perform some named computation on a set of supplied arguments and return a result, is not an especially congenial fit to this application, although it may be a useful tool at a lower level of abstraction in resolving links. Perhaps because the application of linked data has not been widely considered, it is not clear yet what is an appropriate form for the structuring paradigm, or even whether or not the problem has been posed carefully enough to allow one to propose specific structures. This note describes the problem as it has been posed in the distributed data community, and points to a few early attempts to suggest mechanisms; it does not claim to settle the issue.

The interesting requirement in distributed storage of related data lies in those data relations that can be characterized as cross-reference. There are many different applications in which one object might need to contain a cross-reference to another, remote, object. As examples, one might propose to build a distributed hypertext system, a sales reporting system for a corporation that has many branch offices, an office system that allows one worker to incorporate an element of a remote spreadsheet into a local one by reference, a distributed legal case database, with cross-references among cases, and also from state to federal cases and back, or simply an electronic library.

For concrete scenarios consider the specific application of an on-line electronic library (comparable scenarios arise in most of the other application examples.) In the electronic library, the primary scenarios of interest are the following: The maintainer of a document storage service has installed a new document and wants to advertise the new document to clients and clients of clients, and allow search services to store and pass along cross-references (in this application a cross-reference is usually called a “citation”) to the new document. In a related scenario, the client of a search service has completed a search and wants to be able to store a persistent cross-reference to one of the items discovered. Storing

the search query that was used to discover the item is one common suggestion, but that method is not very satisfactory, because that particular search may have returned several items in its result set, or it may have been framed in terms of something unrelated to what makes the document interesting. In addition, since collections grow and shrink, there is no guarantee that the search service will return the same result set for the same query at a future time—especially if the document itself might change in such a way as to cause the query not to find it. In yet another related scenario, the user of a document discovers in it a cross-reference to another document, and wants to make a copy of that cross-reference, in persistent storage for future use. And in a final scenario, a client includes a cross-reference in a document which it then submits to be stored by some storage service, possibly a different one.

In each scenario, some client intends to place the cross-reference in persistent storage somewhere, for presentation to the storage service at an unknown time in the future. When the client eventually presents the cross-reference to the storage service, it hopes to retrieve the cited item.

In an electronic library, such cross-references may be used:

- by an author, in creating a new document, to cite previous, related documents.
- by the librarian of a storage service, in adding a document to a collection, making concrete an author's traditional text citations.
- by a user of a search service, to prepare a list on a personal note pad of discovered documents.
- by a search service as the internal connection between its index and the documents held by a storage service. (e.g., to connect a bibliographic record with the corresponding document).
- by a search service, as the method of telling the client how to obtain the item from a storage service.
- by a client, to pass along to another client
- by an author, to cite anchor (that is, identified internal) points within another document.

The following simple scenario perhaps captures best of all the required structuring behavior: At the application level, a user has brought up a document in a display window, and upon browsing through it noticed that it mentions another document. The browser provides as a feature that the user be able to point to the cross-reference with the mouse, click, and expect to see the cited document appear on the screen in another window.

There are a number of interesting constraints on the solution, at least in the electronic library application. Some of the other applications have analogous constraints:

- The storage service to which the cross-reference is eventually presented may be different from the one that provided the original cross-reference—it may be a backup system, or even a competitive provider.
- The interval between creation and use of the cross-reference can be quite long, perhaps decades, during which time the storage service may have been upgraded, relocated, merged with other storage services, and placed under different administrative control.
- Cross-references will persist long enough that the system that is intended to resolve them will become obsolete.

- Between creation and use of the cross-reference, the cited document may have been deleted, updated, or superseded. Something graceful should happen if multiple versions are available. If the “document” is actually a dynamic piece of data such as a current stock quote, perhaps something different, yet graceful, should happen.
- Between creation and use of the cross-reference, the organization of the library may have been revised, and the target document may now be classified differently.
- Between creation and use of the cross-reference, the physical and low-level logical configuration of the storage server may have been revised, and the target document may be in a different directory or on a different physical volume.
- Between creation and use of the cross-reference, the storage representation of the target document may have been discovered to be defective, and restored from a backup copy.
- The initial discovery mechanism may identify only the document; a later discovery process may identify anchor points of interest.
- The user who presents the cross-reference may or may not be authorized to obtain the document.
- A single document may by indexed by several different search services that are under different administrations.
- In response to presentation of a cross-reference, a storage server may, rather than delivering the desired document, instead return another cross-reference.
- If a client makes inquiries of several different search services, it would like to merge the several responses, which requires that it be possible to figure out which of the returned cross-references are duplicates.

As can be seen from this laundry list of constraints, the requirements on cross-references among distributed data objects read more like the list of requirements for a sophisticated name service than they are like the requirements on an RPC service.

It would appear that at the minimum, a cross-reference internally must be composed of at least two components. The first would be an identifier, perhaps to be presented to a name server, that allows the client to discover an appropriate and current server name, port, and protocol to use, and to verify upon connection that the service at the other end is the intended one. A second component probably is a specific object identifier that server is expected to recognize. Beyond that, one moves into the realm of speculation. There might be an expiration date after which the server doesn't guarantee to honor the cross-reference, and perhaps also a backup query, which might be useful in identifying the object after the expiration date (or in the case of some other failure) of the original cross-reference. Finally, one might want to include some kind of check data to verify that the object retrieved is actually the one that was previously cited. Another potential component, whose rationale is much less clear, is the identity of an application that knows how to interpret the stored object, and that should be launched in conjunction with the arrival of a response from the server, or perhaps spliced into the path between the client and the server.

The details of how such a cross-reference might be engineered, so that it can be stored, passed from client to client, and in the end be recognizable by the server, are an interesting design challenge. Several projects have run up against the challenge, and have suggested various strategies that solve parts of the problem. At least six somewhat different ideas are extant:

- Tim Berners-Lee has proposed the cross-reference scheme used in his World-Wide Web. This proposal is moderately complete, but it concentrates mostly on developing a syntax that can be parsed by a computer and also read by a person. It takes the view that it should be possible to create a document identifier that is both unique and perpetually valid.
- Clifford Lynch, to stimulate discussion of the topic within the Coalition for Networked Information developed a list of requirements, and for each some observations about mechanics that might address that requirement.
- Brewster Kahle has proposed the cross-reference scheme used in his Wide-Area Information Service.
- F. H. Ayers has made a proposal for a universal standard book number.
- Theodor Nelson, for the Xanadu® hypertext system, proposed a universal, hierarchical document numbering scheme with provision for versions and internal anchor points. It covers several of the requirements mentioned earlier by assuming complete homogeneity among the linked items.
- Apple Computer, in the System 7 Alias Manager for the Macintosh, has worked out a sophisticated system for linking files within a Macintosh and across a network of cooperating machines. The Alias Manager uses a combination of symbolic relative and absolute path names as well as unique file, volume, and system identifiers, to maintain links in the face of renaming, hierarchy restructuring, and restoration from backup.

Each proposal addresses one or more parts of the problem, but none covers the entire range of requirements. More important, the discussions by Lynch and by Kahle are characterized by mentions of requirements not met, and possible alternative approaches, together with questions about whether or not the requirements are real. Interestingly, almost as if to recursively emphasize the need for a solution to this problem, most of the current literature on the subject is not found in traditional journals, reports, or libraries, but rather is found only on-line in various repositories within the internet.

Conclusion

As mentioned at the outset, this note has only described the problem, not proposed any solutions; even this description is not likely to resemble the one that will someday seem obvious.

Acknowledgements

Ideas and suggestions came from discussions with Mitchell Charity, Tim Berners-Lee, Andrew Birrell, Dave Redell, Paul McJones, and Ron Weiss.

Bibliography

Tim Berners-Lee, Jean-Francois Groff, and Robert Cailliau. “Universal Document Identifiers on the Network.” CERN, February 1992. On-line location: info.cern.ch:/pub/www/doc/udi1.ps

Clifford Lynch. “Workshop on ID and Reference Structures for Networked Information.” 24 October 1991. (Call for participation. On-line location: CNI-ARCH listserv mailing list at uccvma.bitnet. Also found in *WAIS-discussion digest* #33, 27 November, 1991.)

Brewster Kahle. “Document Identifiers, or International Standard Book Numbers for the Electronic Age.” Version 2.2, September 1991. On-line location: quake.think.com:/pub/wais/doc/doc-ids.txt

F. H. Ayres. “The Universal Standard Book Number (USBN): why, how and a progress report.” *Program: Automated Library and Information Systems 10*, 2 pp. 75-80. (London: Association for Information Management: April, 1976)

Theodore H. Nelson. *Literary Machines, Edition 87.1.* (San Antonio, Texas: Theodor H. Nelson: 1987)

Apple Computer. “Alias Manager,” *Inside Macintosh Volume VI*, chapter 27. (New York: Addison-Wesley: 1991)