To appear in *The Internet as a Large-Scale Complex System*, Kihong Park and Walter Willinger (Editors), published by Oxford University Press as part of Santa Fe Institute series, 2002

Content Networks: Taxonomy and New Approaches

H. T. Kung
Division of Engineering and Applied Sciences
Harvard University, U.S.A.
C. H. Wu
Institute of Information Science
Academia Sinica, Taiwan

Abstract

In this article we describe a taxonomy for content networks and suggest new architectures for such networks. In recent years, many types of content networks have been developed in various contexts, including peer-to-peer networks [8][12][16][27][31][37][49], cooperative Web caching [3][47], content distribution networks [2][7][10], subscribe-publish networks [6][52], and content-based sensor networks [14][21][22]. For each context, there have been numerous architectural approaches with various design objectives. Our taxonomy attempts to formulate a design space for both existing and future content networks and to identify design points of interest. The proposed new content networks, called semantic content-sensitive networks, offer desirable features such as support for content-proximity searches and the use of small routing tables.

I Introduction

A content network is an overlay IP network [4][38] that supports content routing, that is, messages are routed on the basis of their contents rather than the IP address of their destination. Permanent binding of contents to hosts will no longer be necessary to provide access to the contents. Nodes of the overlay network, called network or content nodes, route messages and may also store contents. By using overlay networks, content networks have the flexibility to customize their topologies to meet specific application needs and performance objectives [39][46][60].

Content networks can be attractive for a number of reasons. At any given time, a piece of content and its copy may be freely placed at or moved to any network node to improve content availability, minimize access time, support source anonymity, etc. Because content routing reflects the content of a message, a content network allows properly provisioned links to facilitate the routing of contents that belong to classes deemed valuable. In addition, security can be based on content rather than on site [8][52][53].

In this article we describe a taxonomy for content networks, illustrate various designs in the taxonomy with existing or proposed content networks, indicate strengths of different design approaches, and introduce a new class of content networks that perform "semantic aggregation and content-sensitive placement" of contents. This class corresponds to a design point in our taxonomy that has not been not sufficiently exploited in the literature even though, as we point out, it enjoys certain desirable features, such as support of semantic proximity searching, not shared by other architectures.

This article has the following outline. Section II describes a classification scheme for content networks based on two dimensions: content aggregation and content placement. Section III discusses the properties and presents examples of the four types of content networks in the proposed taxonomy. Sections IV and V give examples of the new type of content networks that use semantic aggregation and content-sensitive placement of content. Section VI illustrates that content networks that perform semantic aggregation can use both content-sensitive and content-oblivious placement of contents, thus offering the strengths of both schemes. Finally, Section VII summarizes and concludes the article.

II Two Classifying Dimensions for Content Networks

We classify content networks on the basis of their attributes in two dimensions:

(1) Content aggregation: semantic vs. syntactic, and
(2) Content placement: content-sensitive vs. content-oblivious.

In Section III.2 these dimensions are used to classify many existing or proposed content networks.

Dimension (1) relates to the use of content aggregation for the purpose of placement and routing. These functions can be performed either on a content-group or on an individual content basis. As described below, using content aggregation, a content network can scale up to massive amount of content. Dimension (2) relates to the placement of contents on network nodes. The placement strategy affects optimization of content routing and the size of routing tables. These two dimensions are orthogonal, in that design choices in one do not necessarily dictate those in the other.

II.1 Content Aggregation: Semantic vs. Syntactic

Content aggregation is a process of assigning individual contents to content groups. The process can be considered to consist of two steps. The first, "aggregation mapping," maps individual contents to values in some value space. The second, "aggregation grouping," groups individual contents on the basis of those mapped values.

Content aggregation can be "semantic" or "syntactic," and content networks that use semantic or syntactic aggregation are called *semantic* or *syntactic content networks*, respectively.

A *Semantic Aggregation*

For semantic aggregation, individual contents are mapped to values that have "meaning" with respect to some external system, and the contents are grouped according to their mapped values in the context of that external system. Consider, for example, an aggregation that uses an animal taxonomy as its external system. In the first aggregation step, contents related to cats and dogs are mapped to the values "cat" and "dog," respectively. In the second step, because cats and dogs are both mammals, the contents are grouped together as the mammal aggregate.

When semantic aggregation is used, the contents grouped in the same aggregate all satisfy certain common features. For example, the mammal aggregate contains contents related to animals that share the biological feature of breast-feeding. Thus, contents in the same aggregate may be characterized, related, or compared in terms of shared features in the associated external system. An interesting consequence is that a semantic content network can allow semantic-approximate or proximity searching [11][17][25]. That is, contents that are semantically related to a given piece of content can be found by searching through network nodes that contain an aggregate to which that content belongs. For example, a search through the mammal aggregate will find contents related not only to dogs and cats but also to mice and elephants, and so on.

B *Syntactic Aggregation*

For syntactic aggregation, the mapped values of contents do not belong to any external system of interest. Consider, for example, an aggregation mapping that is a hash function to be applied to a syntactic identifier of a piece of content, such as its file name. In this case, the mapped value of a piece of content is just a bit-string, which does not reveal the meaning of the content in any useful way. From the bit-string value resulting from a hash calculation, one cannot tell the nature of the content. Furthermore, there is no a priori external system in which comparing bit-string values would make sense. Nevertheless, in this case, one can still group contents on the basis of their hashed bit-string values, as in some peer-to-peer networks [8][12][27]. But contents in the same content aggregate will not share any common features or meaning derived from any external system of interest to applications. This type of aggregation is thus called syntactic, because it lacks semantic meaning.

Because the contents of the same aggregate are not related according to any external criteria of interest, content grouping based on syntactic aggregation can improve the scalability of a content network without revealing the nature of the contents in a group [12][27][37][49]. Syntactic aggregation can be useful when content anonymity is a design objective [8].

Note that a content network may not perform any content aggregation at all, i.e., content placement and routing are both performed on an individual-content basis. We use the convention that content networks that perform no aggregation are considered syntactic networks.

II.2 Content Placement: Content-sensitive vs. Content-oblivious

The placement of content in a content network can be either "content-sensitive" or "content-oblivious." That is, the location where a piece of content or a content group is placed can be either a function of the content or independent of it.

A *Content-Sensitive Placement*

For content-sensitive placement, the location of a piece of content or a content group in a network is a function of the content. Both semantic and syntactic content networks can employ content-sensitive placement. Consider, for instance, a semantic content network with a tree topology that aggregates contents based on an animal taxonomy. Suppose that content placement follows the rule that contents related to dogs can be placed only in a certain subtree reserved for dog-related contents. This, then, is an example of a semantic content network that uses content-sensitive placement. See Sections IV and VI for detailed examples of semantic content networks using content-sensitive placement of content.

Consider next an example of a syntactic content network with a hypercube topology. Suppose that contents are aggregated on the basis of hashed values of their file names and that the placement of the content also is based on these values. Thus, all the contents with the same hashed values are placed at the same network node [12][27]. This is an example of a syntactic content network that uses content-sensitive placement.

With content-sensitive placement, content routing can be made to satisfy specific properties. Consider again the syntactic content network with the hypercube topology, mentioned above. Content-sensitive placement can ensure that, given the hashed values of a piece of content, that content can always be reached from any node through a fixed route determined by those values [12][27].

For semantic content networks, content-sensitive placement can be especially useful. For example, sports contents can be placed on a sports subnet, basketball contents on a subnet of the sports subnet, NBA contents on a subnet of the basketball subnet, etc. In this sense, the topology hierarchy matches the content hierarchy.

Matching topology and content hierarchy yields a number of advantages, some of which are listed here:

- Efficient processing of search queries. For example, queries for content related to the NBA can first be forwarded over the links leading to the sports subnet, then those to the basketball subnet, and, finally, the NBA subnet.
- Small routing tables. The content-routing table of a node can be as small as the number of content groups at the corresponding content layer, independent of the total number of contents in these groups. Consider, for example, a network node that forwards queries over the link that leads to the basketball subnet. Its content-routing table will need to contain only a few dozen types of sports, such as basketball, football, tennis, golf, etc.
- Semantic-approximate or proximity search. For example, while at a network node that contains basketball-related contents, a search process can consult all these contents beyond those of the NBA, such as information about other basketball leagues.
- Properly provisioned network paths for content access. Because the route to any given content from the root of the topology hierarchy is fixed, the underlying path can be properly provisioned to facilitate access to the content. For example, if contents related to the NBA are popular, then high-bandwidth pipes or shortcut links (see Section IV.4) may be allocated between the sports subnet and the NBA subnet. This is in contrast to today's Internet, where bandwidth or link provision for content access can be difficult. For example, in an autonomous system of the Internet, routes are computed using cost metrics based on hop counts with no concern for content.

In content-sensitive placement, the original content may be mirrored at a number of network nodes to facilitate access to the content by nearby nodes [12][27]. Content-sensitive placement refers to the restriction that the placement of the original contents needs to be content-sensitive. This restriction does not necessarily apply to copies of an item of content, which may be stored at mirror sites or on cached servers.

B *Content-oblivious Placement*

For content-oblivious placement, content groups can be placed anywhere without consideration of the content. Because the content can be at any place, the network will need either to learn or to find routes to reach them. To support this, the network may rely on a centralized server to maintain locations of content. In this case, network nodes possessing the content may register their content and locations to the server. A node requesting the content may first query the server for their locations, and then send the request to the locations [31].

Several decentralized schemes may also be used here. For example, nodes that contain contents or their copies can periodically advertise possession of these contents to peering nodes [3][35]. On the basis of such advertisements, the peering nodes build or update their content routing tables for reaching these contents and also advertise their availability in being used by other nodes to reach the contents. The same procedure is repeated at all nodes, when they receive such advertisements from their peers. Many variations on these schemes are possible, including, for example, schemes based on summary vectors, such as the Bloom filters used in cooperative Web caching [3]. Note that when the contents are network IP addresses, the basic scheme is the traditional distance vector algorithm used in IP routing [35].

Alternatively, nodes that request contents can advertise their queries to their peers. Upon receiving a query, a node can reply that it possesses the requested content or, if it does not, relay the request to its peers [8][16].

In content-oblivious placement, a content network allows unconstrained placement of contents on network nodes with no regard for the contents or for the network topology. This convenience is at the expense of transmitting content advertisements or query messages and requiring relatively large routing tables. That is, content-oblivious placement exchanges routing overheads and routing table size for convenient network topology management and content placement.

III Taxonomy for Content Networks

There are four types of content networks based on the two-dimensional classification scheme (see Section II)]. For a summary, see Table 1. Each type corresponds to one of the following four policies regarding treatment of content:

A. Syntactic aggregation and content-oblivious placement;

B. Syntactic aggregation and content-sensitive placement;

C. Semantic aggregation and content-oblivious placement; and

D. Semantic aggregation and content-sensitive placement.

	Syntactic aggregation of content	Semantic aggregation of content
Content-oblivious placement of content	Type A: Syntactic content-oblivious network	Type C: Semantic content-oblivious network
Content-sensitive placement of content	Type B: Syntactic content-sensitive network	Type D: Semantic content-sensitive network

Table 1: Four types of content networks based on the classifying dimensions in Section II.

III.1 Properties of Content Networks

From our discussion in Section II, we can infer properties for each type of content network. For example, content networks that perform syntactic aggregation of their contents can naturally support content anonymity, because no semantic information about the contents is used. Content networks that perform semantic aggregation of their contents can facilitate content proximity searches, because contents with similar meaning are placed together in the network. Content networks that use content-sensitive placement of their contents can support a massive amount of individual contents. This is because the size of the routing tables will depend on the number of content groups a node handles, independent of the total amount of content in the network. Finally, content networks that use content-oblivious placement can be highly fault-tolerant, because the contents and their copies can be placed anywhere and routes to their current locations are learned dynamically.

III.2 Examples of Content Networks

It appears that most of the content networks now deployed belong to types A, B, and C. Type D has not received much attention in the literature. This section provides examples of content networks of all four types. Additional examples of type D content networks are described in Sections IV and V.

Type A   Syntactic Content-oblivious Network

A simple example of this type of content network is one related to Web proxy servers [3][47]. In this case, the contents are URLs, and network nodes are a set of Web proxy servers. Because no content aggregation is performed, this is a syntactic content network. Furthermore, because any given content can reside at any of the proxy servers, it is a content network that uses content-oblivious placement. A similar example is a set of mirror sites that host specific contents [2][7][10].

Many existing content-delivering systems that rely on centralized servers to locate contents without aggregating them can be viewed as syntactic content-oblivious networks. In such systems, the contents can be placed on any node disregarding their meaning, and their locations are registered in a centralized server. These systems include search engines, such as Google [18] and Yahoo [59], where Web pages are contents hosted on Web server linked by the URLs with which the search engines respond to queries. A set of Web services [51][54], described with WSDL [58], communicated with SOAP [57] and registered at the UDDI servers [50], may also be viewed as syntactic content-oblivious networks.

Most existing peer-to-peer (P2P) systems that do not aggregate contents, such as Napster, Gnutella, and Freenet, are syntactic content-oblivious networks. The contents in these systems may be placed on any node, independent of the location of other contents. The contents are searched via centralized servers (e.g., Napster [31][32]) or through query flooding (e.g., Gnutella [9][16] and Freenet [8][15]). These systems usually incur serious scalability problems [1] [26][41][48].

Consider P2P systems that use query flooding. In the case of Gnutella, when a node receives a query that it cannot serve owing to the absence of the requested content from the local store, the node will forward the query to other nodes in a breadth-first manner [9][16]. A node will stop forwarding a query that was previously received or whose time-to-live counter has expired. In contrast, in the case of Freenet, depth-first search is used [8][15]. When receiving a query from its parent peer node, a node will serve the request if it has the requested content in the local store. Otherwise, it will forward the query to the one child that can most likely serve the query. If this child fails to serve the query, then the node will try another child with the next highest potential for success in serving the query. If none of the children succeeds, the node will notify its parent peer node that it cannot serve the request.

SIENA [6], a subscribe-publish network that provides event notification services, is another example of syntactic content-oblivious networks. In SIENA, the client, called a subscriber or publisher, issues a subscription to or a notification of events to a nearby server, and a set of interconnected servers that advertise and multicast the events to interested parties. In this case, the contents are subscriptions and notifications, and the network nodes are subscribers, publishers, and forwarding servers. A notification is pushed to interested subscribers that have previously advertised related subscriptions. To match notifications and subscriptions, a network node may use a filter or pattern to match a set of events. Because content is not aggregated and identical or similar events could be issued by any subscriber or publisher, SIENA is a syntactic content-oblivious network. Similarly, Publius [52], a system for Web publishing, is a syntactic content-oblivious network.

The subscribe-publish model has been used in sensor networks such as SCADDS [14][21] and SPIN [22]. These are syntactic content-oblivious networks, in which the contents are detected data or signals and the content nodes are sensors.

Type B Syntactic Content-sensitive Network

Some P2P systems that support distributed hashing tables [39] have adopted syntactic aggregation and content-sensitive placement strategies in order to avoid using nonscalable schemes, such as centralized servers or flooding query, described above. Several examples are mentioned here.

PAST is a persistent P2P storage utility that uses a hypercube-based routing scheme, called Pastry, to route contents [12][42][43]. In Pastry, every peer node is assigned a 128-bit node identifier (NodeId), derived from a cryptographic hash of the node's public key. Every file is assigned a quasi-unique 160-bit file identifier (FileId) that corresponds to the cryptographic hash of the file's textual name, the owner's public key, and a random salt. Nodes in Pastry are connected using a hypercube topology, in which adjacent nodes share some common address prefixes. Pastry routes a message concerning a given FileId toward the node whose NodeId is numerically closest, among all live nodes, to the 128 most significant bits of the FileId. This means that contents are aggregated syntactically according to the FileId and placed at the numerically closest node.

Like Pastry, OceanStore [27] uses a hypercube algorithm, called Tapestry [62], which is a variation on a randomized hierarchical distributed data structure [36] that can deterministically locate an object. In Tapestry, every server is assigned a random (and unique) NodeId, and every object is identified by a globally unique identifier (GUID), which is a secure hash of the owner's key and some human-readable name. Objects are syntactically aggregated and content-sensitively placed in a manner similar to the process in Pastry. OceanStore is a highly redundant, persistent storage system. Before invoking Tapestry, OceanStore will first use a probabilistic algorithm called attenuated Bloom filters to track and locate objects. Each node in OceanStore maintains an array of Bloom filters where the *i*th Bloom filter is the union of the Bloom filters for all the nodes at distance *i* from the current node. A query is routed to the node whose filter indicates the presence of the object at the smallest distance.

Unlike these examples that use hypercube structures, Chord architecture [49] uses a ring topology to provide scalable peer-to-peer lookup services. Nodes and files are assigned m-bit identifiers computed by a hash function. The nodes form an identifier circle in the order of the the identifiers' numeric values. Given a key such as the hashed identifier of a file name, Chord maps the key onto a node. The file with FileId k will then be assigned to the first node with an identifier equal to or that follows k in the identifier space. For the node with NodeId i and its preceding node with NodeId j in the identifier circle, files with FileId k where $j < k \leq i$ are aggregated and placed at the node with NodeId j.

CAN [37] uses a virtual d-dimensional coordinate space, which may be viewed as a generalization of the one-dimensional identifier scheme above. At any time the entire coordinate space is dynamically partitioned among all the nodes such that each node in CAN owns an individual, distinct zone in the overall space. Every file is represented as a point in a specific zone maintained by some node. A CAN node holds information about a small number of adjacent zones in the space. Using a simple greedy forwarding scheme, the node routes a message to the adjacent zone with coordinates closest to the destination coordinate.

Type C   Semantic Content-oblivious Network

In semantic content-oblivious networks, the contents are aggregated semantically and the resultant content groups can be placed anywhere. Query broadcast or source advertisement is used to search the contents (see Section II.2B).

TRIAD [19] is an example of this type of content network. The contents are mapped to their URLs and grouped on the basis of the domain name portion of the URL. Thus, contents with the same domain name are aggregated into the same group. The URL structure is the "external system" as discussed in Section II.1A. In TRIAD, the Internet Name Resolution Protocol (INRP) performs name lookup, whereas the Name-Based Routing Protocol (NBRP) performs routing advertisement. Like BGP [40], NBRP allows content aggregations to reside anywhere in the content network. In addition, TRIAD proposes to use routing aggregate to reduce the size of name-routing table.

Type D   Semantic Content-sensitive Network

An example of a semantic content-sensitive network is the well known Internet Domain Name System (DNS) [28][29][30]. Consider the DNS service that translates host names or domain names into their IP addresses. In this case, the contents are host names and domain names described in fully qualified domain names. In the DNS lookup service, the contents are semantically aggregated into groups according to existing organizational hierarchies, and the content aggregates are placed at DNS servers that belong to the corresponding organizations. Thus, the DNS network is a content network that performs semantic aggregation and content-sensitive placement of content. The next two sections give further examples of content networks of this type, which use data-driven and static content grouping.

IV Semantic Content-sensitive Network: Using Data-driven Content Grouping

This section describes a semantic content-sensitive network in which the aggregation grouping is data-driven (see Section II.1), in the sense that it will result from a clustering analysis of all the contents present in the network. First, the data-driven grouping is described, and then the construction of the corresponding semantic content-sensitive network.

Note that in Section V, the same method for constructing a content network is used for the case of static grouping of content.

IV.1 Data-driven Content Grouping

This section describes a clustering method for grouping a given set of contents using a clustering tree derived from content vectors associated with the contents. First, the content vectors are described, then the clustering tree, and, finally, the grouping method.

A *Content Vectors*

We assume that the given contents are associated with content vectors that indicate similarity or dissimilarity. The content vector associated with a piece of content can be considered its “mapped value” in the aggregation process (see Section II.1). For example, when the content is a Web page, the associated content vector may be expressed in terms of the frequencies of certain key words in the page. Such vectors are similar to term weighting in the vector space model used in conventional information retrieval systems [44]. For some typical methods of assigning content vectors to contents, refer to [45]. In the rest of Section IV, we use the terms content and the associated content vector interchangeably.

There are two major categories of similarity metrics for content vectors [25]. These are angle-based metrics which use, e.g., the cosine function, and distance-based metrics which use, e.g., the inner-product function. For the discussion below, we assume distance-based metrics.

Figure 1 illustrates a set of two-dimensional content vectors associated with eight pieces of content: LA, NY, NCAA, PGA, Mozart, Bach, Jack, and Helen. These vectors correspond to points in the two-dimensional space. With a distance-based metric the positions of the content vectors in the two-dimensional space reflect their similarity, that is, contents of similar nature are close to each other. For example, LA and NY are similar in the sense that both are cities that host NBA teams, and thus their vectors in the two-dimensional space are close together. It should be clear that the concepts illustrated here generalize to content vectors of dimensions higher than two.

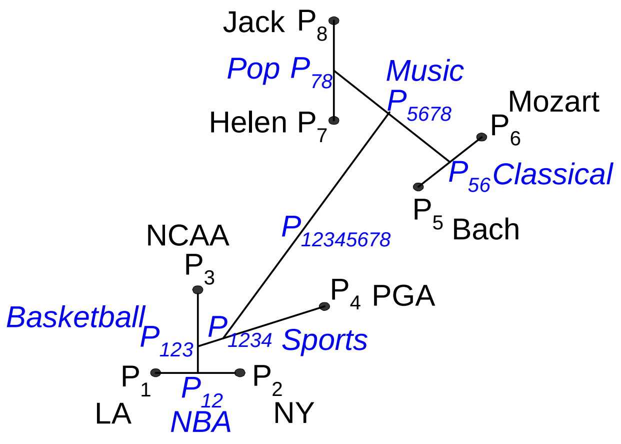

Figure 1: Eight content vectors, P_1, P_2, …, P_8, and their content clustering tree.

B *Clustering Trees*

Given a set of contents, we construct a clustering tree for their content vectors. A number of methods are available in the literature for the construction of clustering trees [20][23][24]. Figure 1 illustrates one of the simplest methods. Using this method, we start by finding a pair of points, P_1 and P_2, that is one of the closest pairs among all pairs of points. Then the two points are connected, and for the rest of the clustering process, they are replaced by a new point P_{12}, which is a midpoint of the segment (P_1, P_2). The process is repeated until all points are connected.

The binary tree that results from the above process of connecting all the content vectors is called a *clustering tree*. Each of its subtrees defines a *cluster* that contains the leaf content vectors of the subtree. We call the root of the subtree the *centroid* of the cluster. Note that in the construction of the clustering tree, each step can be viewed as connecting two centroids and creating a new one. The distance between the two centroids to be connected at each step is nondecreasing as the construction proceeds. Thus, sibling centroids become farther apart or remain at the same distance at the next higher layer of the tree.

A cluster can be represented by its centroid in the sense that the content vectors associated with the cluster are generally closer to this centroid than to its sibling centroid. For example, P_{5678} is the centroid of the cluster consisting of P_5, P_6, P_7, and P_8, and these points are closer to P_{5678} than the sibling P_{1234}.

To reflect the volume of contents in the associated cluster, a centroid might be weighted by moving it toward the centroid of a subtree that has more leaves than the other subtree. For example, in Figure 1 the position of a midpoint reflects the “weights” of the two subtrees, i.e., the amount of content under them. For example, midpoint P_{123} is closer to P_{12} than to P_3 by a factor of two, because the subtree rooted at P_{12} has twice as many content nodes as the subtree rooted at P_3.

C *Content Grouping Based on Content Clustering Tree*

Given a content clustering tree, we can aggregate contents in groups each of which is associated with a centroid. For each internal node of the clustering tree, we use a dividing hyperplane perpendicular to the line connecting to the two centroids of the two subtrees [5][34]. The hyperplane partitions the space of the content vectors into two groups, each containing one of the centroids.

To illustrate the grouping, consider the clustering tree in Figure 2 (a), where the dividing hyperplanes are denoted by dotted lines. As shown, when content vectors are two-dimensional, a dividing hyperplane is a line. Note that there is a hierarchy among groups, in the sense that a group associated with a centroid of a tree contains the two subgroups associated with the centroids of the two subtrees.

A hyperplane can be weighted by moving it toward the centroid of a subtree that has fewer leaves than the other subtree. In Figure 2 (a) the dividing hyperplane corresponding to the root of the clustering tree illustrates this weighting. Note that the weighting direction used here is opposite to that used in weighted centroids described earlier in Section IV.1B. Both weighting heuristics are for the same purpose, that of increasing the amount of content in the same clusters, and will be grouped together by dividing hyperplanes. Other weighting heuristics could also be used to reflect, e.g., expected content access frequencies, rather than number of contents present.

IV.2 Construction of Content Network

Suppose that we are given a set of contents with an associated clustering tree. We can construct a content network with the same topology, using content-sensitive placement of the content.

A *Topology of Content Network*

Each node in the content network corresponds to a node of the clustering tree. Two content nodes are connected in the content network if, and only if, the two corresponding nodes in the clustering tree have the parent-child relationship. Consider, for example, the clustering tree in Figure 2 (a). Figure 2 (b) depicts the content network with the same topology.

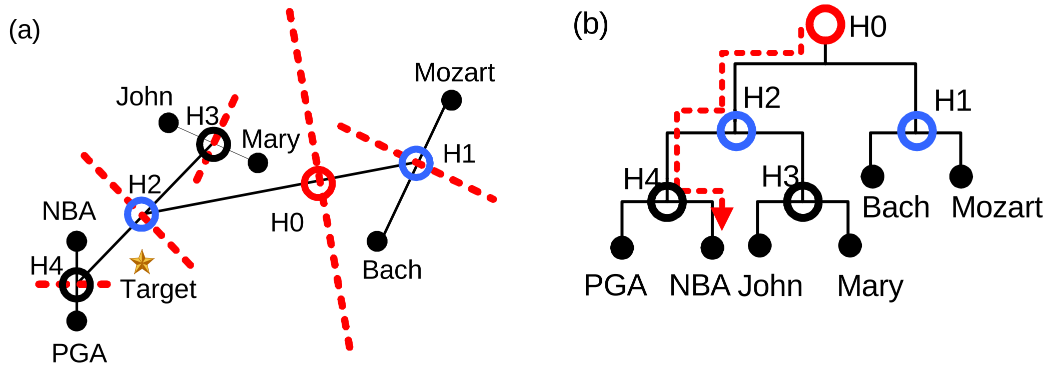

Figure 2: (a) Content clustering tree with dividing hyperplanes constructed using Voronoi diagrams; and (b) content networking using the same tree topology and content-sensitive placement of content.

B *Content-sensitive Placement of Content*

Figure 2 illustrates content-sensitive placement. For a given piece of content, we describe the process of finding the node where the content will be placed. Suppose that the given piece of content has its associated content vector at the location "target" shown in Figure 2 (a). We show how a route leading to the content node NBA, which is the node closest to the "target", is derived. The route will start at the root H0 of the content network. To determine the next hop, we make use of the dividing hyperplane at the corresponding node in the clustering tree. Because the hyperplane partitions the entire space of the content vectors, one of the partitions must contain the given piece of content. Consider the child of the corresponding node in the clustering tree that is in the partition containing the given piece of content. Then next hop will be the node in the content network that corresponds to this child in the clustering tree. Following this method, the route will go to H2 from H0, to H4 from H2, and finally to NBA from H4, as depicted in Figure 2 (b).

IV.3 Properties of the Resulting Content Network

The content network of Figure 2 (b) is a semantic content-sensitive network, because it uses semantic aggregation and content-sensitive placement of content. The network has the advantages of type D content networks (see Section II.2A). It can support content proximity searches while using relatively small routing tables with sizes independent of the total number of contents in the network. For example, the search suffices to go only to the subtree rooted at H4 for contents related to PGA and NBA. The routing table at H1 or H2 need only contain two entries. If contents related to NBA deserve high-bandwidth access, then specially provisioned network paths can be provided from the root to the node NBA.

IV.4 Shortcut Optimization

Shortcut links can be added to provide direct links between content nodes, as illustrated in Figure 3. Using the shortcut links from H0 to H3 and H4 and the existing link from H0 to H1, the content node H0 can directly forward a query to H1, H3, and H4, without going through the intermediate content node H2. The search direction at H0 will now be determined by which of the three sites of Figure 3 (d) has the target content, rather than either of the two sites of Figure 3 (c). To determine the next hop from H0, two dividing hyperplanes, rather than one, will be needed, which can be computed using Voronoi diagrams [5][34].

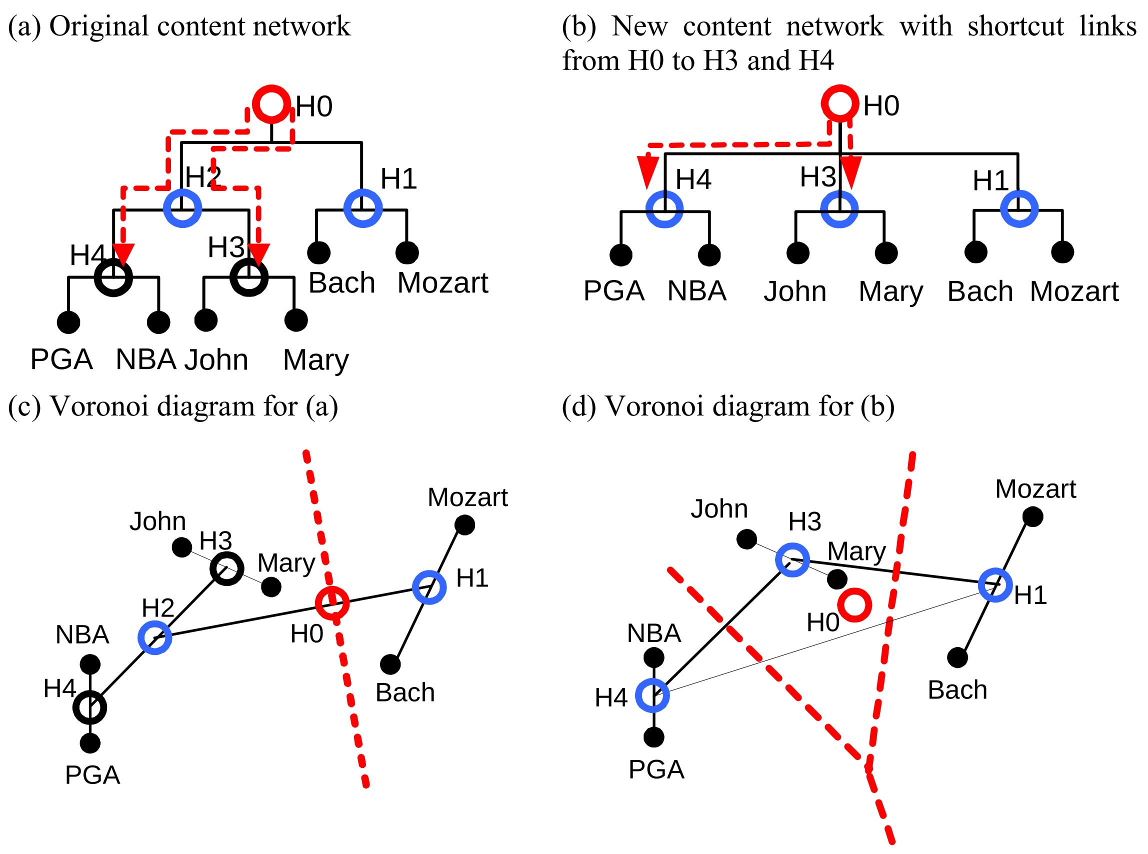
(a) Original content network
H0
H2
H1
H4
H3
Bach
Mozart
PGA
NBA
John
Mary
(b) New content network with shortcut links from H0 to H3 and H4
H0
H4
H3
H1
PGA
NBA
John
Mary
Bach
Mozart
(c) Voronoi diagram for (a)
John
H3
Mary
Mozart
H1
H2
H0
NBA
H4
Bach
PGA
(d) Voronoi diagram for (b)
John
H3
Mary
Mozart
H1
H0
NBA
H4
Bach
PGA

Figure 3. Shortcut optimization. (a) Node H0 recognizes that there is heavy traffic from H0 to both H3 and H4, and there is no traffic destined to H2. (b) H0 creates shortcut links to both H3 and H4, without involving H2. H0 will now direct route to one of the three sites of (d) rather than the two sites of (c).

V Semantic Content-sensitive Network: Using Static Content Grouping

This section describes a semantic content-sensitive network in which the aggregation grouping referred to in Section II.1 is static, that is, the association of a piece of content with a group is static and independent of other contents in the network. This is in contrast to the data-driven grouping discussed in Section IV.

Many existing content-retrieval systems use static hierarchical structures already in place to group the content. These include library catalog systems [13], DNS [28][29][30], and LDAP [61]. In addition, directory-based search engines such as Yahoo [59] and Open Directory [33] use catalogs to classify Web links. In all these systems, subject categories form a hierarchical tree, and every object is classified into a subject category. Objects belonging to the same subject category share certain common semantic meanings.

For these systems, semantic content-sensitive networks can be constructed using exactly the same approaches as discussed in Section IV, except that the cluster-forming portion of the task is no longer needed. For this reason our description of the construction of the content network here will be brief.

Given a static hierarchical content grouping scheme, we can construct a semantic content-sensitive network. The content network is constructed as described in Section IV.2, with the given content hierarchy replacing the clustering tree.

For example, by referring to an external hierarchical taxonomy such as Open Directory [33], we can represent contents and subject categories with XML [55] and RDF [56]. The content network will have the same topology as the taxonomy hierarchy. With contents of the same subject grouped together and placed in the same node, the resultant content network is thus a semantic content-sensitive network.

To process a request related to a given piece of content, we need a method of associating the request with groups that contain the content. One approach, called *self-describing requests,* is to assume that every request message, such as a content insert or a content query, is self-describing in the sense that the query itself includes specific group information. Thus, when a content node receives such a request, it will know where to forward it on the basis of the group information in the request. This approach is used, for example, by DNS, where every resolving query itself specifies its fully qualified domain name [28][29][30]. This allows a node to forward unresolved queries to the proper server responsible for handling domain names in the next level up.

A content network that uses self-describing requests will not directly support a proximity search. Instead, external mechanisms will be used to map a request concerning a piece of content to one concerning a group to which that content belongs. This mechanism is similar to an Internet directory search engine that responds to keyword-based queries with categories or URLs that are deemed to be related to the keywords in the query.

Another approach, called *self-classifying nodes*, is to assume that every content node is capable of mapping a piece of content related to a request to a proper group to which the content belongs. For example, every content node may keep a copy of a taxonomy database and may determine the proper group for any given piece of content. When the node receives a request message for content query or content insert, it consults its local copy of the taxonomy to determine the next hop to forward the request. For this type of content network, a request message no longer needs to be attached with precise taxonomy information describing the content in the request.

Consider Figure 4 for an example of a semantic content-sensitive network that uses self-classifying nodes. Using the content taxonomy in Figure 4 (a), we construct the content network shown in Figure 4 (b). In this network, each subject category is assigned to a content node, every node has a copy of the taxonomy of Figure 4 (a), and the contents are semantically aggregated and placed at proper nodes. When receiving a request to find content near NBA, H0 knows, by consulting its local copy of the taxonomy, that the content belongs to Category *Sports* (H2), and, similarly, H2 knows it belongs to Category *Associations* (H4). Because every node has a copy of the taxonomy and needs to have only routing entries to its child nodes, the network can support content proximity searching with relatively small routing tables.

In addition, shortcut optimization, described in Section IV.4, can be applied here. For example, if content related to NBA deserves high-bandwidth access, specially provisioned network paths can be supplied from the root to the node NBA, or shortcut links added directly from the root to it.

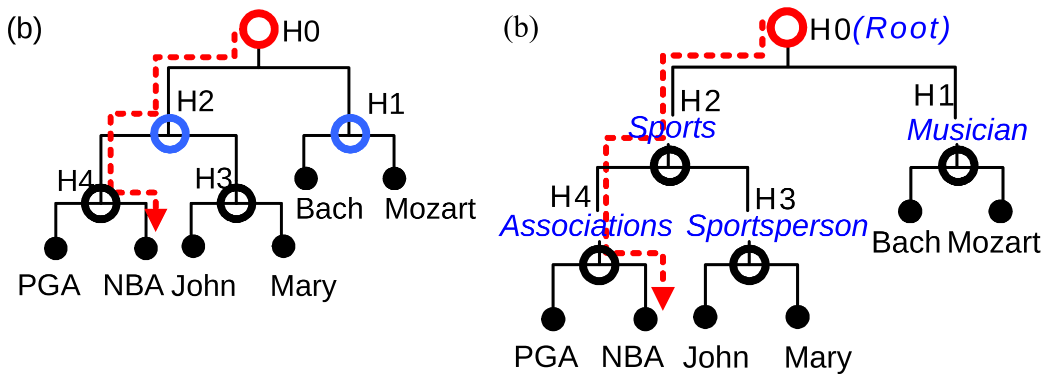

Figure 4: (a) A content taxonomy; and (b) content networking using the same tree topology and using content-sensitive placement of content.

VI Semantic Content Network Using Hybrid Placement

A semantic content network can use content-oblivious and content-sensitive placement for different parts of the network at the same time. This hybrid approach can have the strengths of the content-oblivious scheme such as fault-tolerance and those of the content-sensitive scheme such as small routing tables and content proximity search, as discussed in Section III.1.

Consider, for example, the network model shown in Figure 5, where a set of stub networks is connected through a backbone. There are two types of these stub networks: *content* *stub networks* and *access stub networks*. A content stub network is a semantic content network specializing in some content area such as sports, arts, computers, music, or news, using content-sensitive placement. An access stub network is an access network through which users can connect to the backbone. The overall network is semantic in the sense that contents are semantically grouped and placed in the corresponding content stub networks.

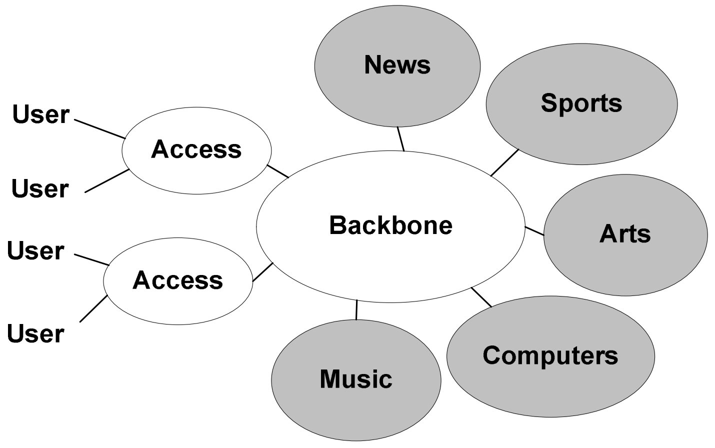

Figure 5. A network model consisting of content and access stub networks and a backbone. Content stub networks are denoted by grayed ellipses. The backbone uses content-oblivious placement of content stub networks, whereas content stub networks use content-sensitive placement of contents within their networks.

Content stub networks use content-sensitive placement of content, which allows the local network administrator to customize content aggregation and placement to support content searches and management. An example of a content stub network can be the network shown in Figure 2 (b) (see Section IV) with its root connected to the backbone.

The backbone uses content-oblivious placement. That is, the root of a content stub network may be placed at any edge node of the backbone, regardless of the content hosted by the content stub network.

The roots of content stub networks will periodically advertise their presence and locations over the backbone, as described in Section II.2B. On the basis of the advertised information, the backbone sets up its content-routing tables. Using these routing tables, the backbone forwards user requests arriving from access stub networks that address contents in a particular area to the root of the corresponding content stub network.

Alternatively, the roots of content stub networks can register their content and locations in a centralized server, such as a UDDI server [50], with protocols such as SOAP [57]. By querying this server, the background can forward user requests to the roots of those content stub networks which are related to the requested content.

The use of content-oblivious placement over the backbone offers important advantages. Content stub networks can be migrated to or replicated at locations where access to a particular type of content is frequent or has a high value. For example, sports information related to soccer and its mirrors could be kept in a variety of content stub networks situated in Europe, where interest in this sport is enormous. As described above the backbone will automatically learn the locations of these content stub networks.

The network model of shown in Figure 5 suggests a deployment strategy for semantic content-sensitive networks. That is, they can initially be deployed as content stub networks.

Finally, we note that the hierarchical network topology of the backbone and stub networks depicted in Figure 5 could allow various combinations of content network types. For example, opposite to the scenario described in this section so far, the backbone could use content-sensitive deployment of content stub networks and stub networks could use content-oblivious placement of contents within their networks. This arrangement can make sense when bandwidth is abundant in content stub networks, but it is scarce in the backbone.

VII Conclusions

Content networks as described in this article are networks in which the addressing and routing of content is based on content rather than on their locations. Thus, in a content network, content is free to move around and be replicated while it remains accessible.

Content networks can address some of the limitations found in current IP networks. For instance, the absence of location addresses on a content network reduces vulnerability to denial of service attacks. Because the location of the content is arbitrary and need not necessarily be associated with a particular server, content providers can be anonymous. Content networks can support new types of client applications that require flexible addressing or no addressing at all, such as peer-to-peer or mobile computing. Content networks can properly provision paths leading to contents deemed to be valuable. As semantic content networks, they support content proximity searches. When content-sensitive placement is used, these networks allow the use of small routing tables to handle massive amounts of content.

In this paper, we have provided a taxonomy that attempts to capture a design space of both current and future content networks. We have identified design points of interest and characterized their individual strengths. In particular, we have described a new type of semantic content network that uses content-sensitive placement of content and argued that these networks possess desirable properties and deserve further study.

Acknowledgments

This research was supported in part by the Air Force Office of Scientific Research Multidisciplinary University Research Initiative Grant F49620-97-1-0382, in part by DARPA under Air Force Research Laboratory AFRL/SNKD F33615-01-C-1983, and in parts by research grants from Microsoft Research, Nortel Networks and Sun Microsystems.

References

[1] Adar, E., and Huberman, B. A., “Free Riding on Gnutella,” Technical report, Xerox PARC, August 2000. Available at http://www.hpl.hp.com/shl/papers/gnutella/Gnutella.pdf.

[2] Akamai Homepage, http://www.akamai.com/

[3] Almeida, J., Broder, A., Cao, P., and Fan, L., “Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol,” in Proc. of ACM SIGCOMM’98, September 1998.

[4] Andersen, D., Balakrishnan, H., Kaashoek, F., and Morris, R., “Resilient Overlay Networks,” in Proc. of the 18th ACM Symposium on Operating Systems Principles, October 2001.

[5] Aurenhammer, F., “Voronoi Diagram - A Survey of a Fundamental Geometric Data Structure,” ACM Computing Surveys, 23(3):345-405, September 1991.

[6] Carzaniga, A., Rosenblum, D., and Wolf, A., “Design and Evaluation of a Wide-Area Event Notification Service,” ACM Transactions on Computer Systems, 19(3):332-383, August 2001.

[7] Cisco, “Cisco Content Networking Architecture,” http://www.cisco.com/go/cdn

[8] Clarke, I., Sandberg, O., Wiley, B., and Hong, T. W., “Freenet: A Distributed Anonymous Information Storage and Retrieval System,” in Proc. of ICSI Workshop on Design Issues in Anonymity and Unobservability, 2000.

[9] Clip2, “The Gnutella Protocol Specification v0.4,” http://dss.clip2.com/ GnutellaProtocol04.pdf

[10] Digital Island Homepage, http://www.digitalisland.net/[11] Dean, J., and Henzinger, M., “Finding Related Pages in the World Wide Web,” in Proc. of the 8th World-Wide Web Conference, May 1999.

[12] Druschel, P., and Rowstron, A., “PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility,” in Proc. of HotOS VIII, May 2001.

[13] Dublin Core Metadata Initiative Homepage, http://dublincore.org/

[14] Estrin, D., Govindan, R., Heidemann, J., and Kumar, S., “Next Century Challenges: Scalable Coordination in Sensor Networks,” in Proc. of ACM MOBICOM'99, August 1999.

[15] “Freenet Protocol 1.0 Specification,” http://freenetproject.org/index.php?page=protocol/

[16] Gnutella Homepage, http://gnutella.wego.com/

[17] Goldman, R., Shivakumar, N., Venkatasubramanian, S., and Garcia-Molina, H.,

"Proximity Search in Databases," in Proc. of the 24th International Conference on Very Large Data Bases, August 1998.

[18] Google Homepage, http://www.google.com/

[19] Gritter, M., and Cheriton, D. R., "An Architecture for Content Routing Support in the Internet," in Proc. of the 3rd USENIX Symposium on Internet Technologies and Systems, March 2001.

[20] Han, E.-H., and Karypis, G., "Centroid-Based Document Classification: Analysis and Experimental Results," in Proc. of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), September 2000.

[21] Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D., and Ganesan, D., "Building Efficient Wireless Sensor Networks with Low-Level Naming," in Proc. of the 18th ACM Symposium on Operating Systems Principles, October 2001.

[22] Heinzelman, W., Kulik, J., and Balakrishnan, H., "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," in Proc. of ACM MOBICOM'99, August 1999.

[23] Jain, A., and Dubes, R., "Algorithms for Clustering Data," published by Prentice-Hall, 1988.

[24] Jain, A., Murty, M., and Flynn, P., "Data Clustering: A Review," ACM Computing Surveys, 31(3):264-323, September 1999.

[25] Jones, W., and Furnas, G., "Pictures of Relevance: A Geometric Analysis of Similarity Measures," Journal of the American Society for Information Science, 38(6):420-442, November 1987.

[26] Jovanovic, M., "Scalability Issues in Large Peer-to-Peer Networks - A Case Study of Gnutella," University of Cincinnati Technical Report 2001. Available at http://www.ececs.uc.edu/~mjovanov/Research/paper.html

[27] Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Weimer, W., Wells, C., and Zhao, B., "OceanStore: An Architecture for Global-Scale Persistent Storage, in Proc. of ASPLOS'00, November 2000.

[28] Mockapetris, P., "Domain Names—Concepts and Facilities," RFC 1034, November 1987.

[29] Mockapetris, P., "Domain Names—Implementation and Specification," RFC 1035, November 1987.

[30] Mockapetris, P., and Dunlap, K., "Development of the Domain Name System," ACM SIGCOMM Computer Communication Review, 18(4):123-133, August 1988.

[31] Napster Homepage, http://www.napster.com/

[32] “Napster Messages,” http://opennap.sourceforge.net/napster.txt

[33] Open Directory Project Homepage, http://dmoz.org/

[34] Okabe, A., Boots, B., Sugihara, K., Chiu, S. N., and Okabe, M., “Spatial Tessellations: Concepts and Applications of Voronoi Diagrams,” second edition, published by John Wiley and Sons, July 2000.

[35] Perlman, R., “Interconnections: Bridges, Routers, Switches, and Internetworking Protocols,” second edition, published by Addison-Wesley, 2000.

[36] Plaxton, C., Rajaraman, R., and Richa, A., “Accessing Nearby Copies of Replicated Objects in a Distributed Environment,” in Proc. of ACM SPAA’97, June 1997.

[37] Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S., “A Scalable Content-Addressable Network,” in Proc. of ACM SIGCOMM’01, August 2001.

[38] Ratnasamy, S., Handley, M., Karp, R., Shenker, S., “Topologically-Aware Overlay Construction and Server Selection,” in Proc. of IEEE INFOCOM’02, June 2002.

[39] Ratnasamy, S., Shenker, S., and Stoica, I., “Routing Algoriths for DHTs: Some Open Questions,” in Proc. of the First International Workshop on Peer-to-Peer Systems, March 2002.

[40] Rekhter, Y., and Li, T., “A Border Gateway Protocol 4 (BGP-4),” RFC 1771, March 1995.

[41] Ritter, J., “Why Gnutella Can’t Scale. No, Really,” http://www.darkridge.com/~jpr5/.

[42] Rowstron, A., and Druschel, P., “Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems, ” in Proc. of Middleware Conference, November 2001.

[43] Rowstron, A., and Druschel, P., “Storage Management and Caching in PAST, a Large-Scale, Persistent Peer-to-Peer Storage Utility,” in Proc. of the 18th ACM Symposium on Operating Systems Principles, October 2001.

[44] Salton, G., Wong, A., and Yang, C. S., “A Vector Space Model for Automatic Indexing,” Communications of the ACM, 18(11):613-620, November 1975.

[45] Salton, G., and Buckley, C., “Term-Weighting Approaches in Automatic Text Retrieval,” Information Processing and Management, 24(5):513-523, 1988.

[46] Saroiu, S., Gummadi, P., and Gribble, S., “Exploring the Design Space of Distributed and Peer-to-Peer Systems: Comparing the Web, TRIAD, and Chord/CFS,” in Proc. of the First International Workshop on Peer-to-Peer Systems, March 2002.

[47] Squid Web Proxy Cache Project, http://www.squid-cache.org/

[48] Sripanidkulchai, K., "The Popularity of Gnutella Queries and Its Implications on Scalability," http://www.cs.cmu.edu/~kunwadee/research/ p2p/paper.html.

[49] Stoica, I., Morris, R., Karger, D., Kaashoek, F., and Balakrishnan, H., "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in Proc. of ACM SIGCOMM'01, August 2001.

[50] Universal Description, Discovery, and Integration (UDDI) Specification, available at http://www.uddi.org/

[51] Vaughan-Nichols, "Web Services: Beyond the Hype," IEEE Computer Magazine, 35(2):18-21, February 2002.

[52] Waldman, M., Rubin, A., and Cranor, L., "Publius: A Robust, Tamper-Evident, Censorship-Resistant Web Publishing System," in Proc. of the 9th USENIX Security Symposium, August 2000.

[53] Wang, C., Carzaniga, A., Evans, D., and Wolf, A., "Security Issues and Requirements for Internet-Scale Publish-Subscribe Systems," in the 35th Proc. of Hawaii International Conference on System Sciences, January 2002.

[54] Web Services Interoperability Organization Homepage, http://www.ws-i.org/

[55] World Wide Web Consortium, "Extensible Markup Language (XML) Specifications," http://www.w3c.org/XML/

[56] World Wide Web Consortium, "Resources Description Framework (RDF) Specification," http://www.w3c.org/RDF/

[57] World Wide Web Consortium, "Simple Object Access Protocol (SOAP) Specification," http://www.w3c.org/TR/SOAP/

[58] World Wide Web Consortium, "Web Services Description Language (WSDL) Specification," http://www.w3.org/TR/wsdl

[59] Yahoo Homepage, http://www.yahoo.com/

[60] Yang, B., and Garcia-Molina, H., "Comparing Hybrid Peer-to-Peer Systems," in Proc. of the 27th International Conference on Very Large Data Bases, September 2001.

[61] Yeong, W., Howes, T., and Kille, S., "Lightweight Directory Access Protocol," RFC 1777, March 1995.

[62] Zhao, B., Kubiatowicz, J., and Joseph, A., "Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing," U. C. Berkeley Technical Report UCB/CSD-01-1141, April, 2001.