PERFORMANCE MANAGEMENT OF PEER-TO-PEER DISTRIBUTED HASH TABLES

Guillaume Doyen[1]
LORIA - University Henry Poincaré
Guillaume.Doyen@loria.fr

Emmanuel Nataf[1]
LORIA - University of Nancy 2
Emmanuel.Nataf@loria.fr

Olivier Festor[1]
LORIA - INRIA Lorraine
Olivier.Festor@loria.fr

[1] *The Madynes research team*
LORIA, 615 rue du Jardin Botanique
54602 Villers-lès-Nancy, France

Abstract P2P networking is a distributed model where entities play both the client and server role. One major problem addressed in this model is the discovery, searching and routing in a dynamic distributed environment. Among the different envisaged solutions, Distributed Hash Tables (DHT) are very promising. They allow the build of robust content addressable networks. Despite good theoretical performance properties, infrastructures which implement the model need a performance management framework able to monitor them in case of a concrete deployment. In this article we propose a generic performance management information model for DHTs. Our contribution uses a standard management approach based on the Common Information Model (CIM) Metric model.

Keywords: Peer-to-peer, network management, performance, information model, Common Information Model (CIM), distributed hash tables.

1. Introduction

Nowadays, P2P networking is an emerging model that extends the limits of the client/server one. Indeed, applications built over it present a better scala-

bility, load balancing and fault tolerance. Although "wild" P2P applications, like that of illegal files sharing, don't want any management infrastructure at all, enterprise ones, used for critical applications, need a management infrastructure. For example, enterprises, administrations or universities may want to deploy P2P applications for several purposes like the distribution of networked file systems, including the replication of data, or the use of distributed collaboration tools for project that count remote participants. In this context, the need for a management framework is obvious in order to ensure services level agreements in value-added applications.

One of the major problems P2P infrastructures have to face concerns the discovery of resources and the routing of messages. The main cause is that P2P applications are composed of versatile entities that form a dynamic environment and may not use any central server for resource location. Among the different solutions envisaged to address the discovery and routing problem, the use of Distributed Hash Tables (DHT) proves to be a very efficient solution which enables the construction of robust content addressable networks [Fraigniaud and Gauron, 2003].

These frameworks offer interesting theoretical performances in terms of average path length, load balancing and consistency. Nevertheless, there is actually no way to monitor the performance they announce in case of real deployments. In this paper, we propose a performance management instrumentation model for DHTs. It extends a CIM generic model for P2P networks and services [Doyen et al., 2004a] that encompasses the functional, organizational and topological aspects of this networking model. Our performance model is generic and can be applied to any existing DHT infrastructure in order to monitor its performance.

This paper is organized as follows: Section 2 presents the generic P2P information model we have designed. Section 3 focuses on the DHT model and the existing theoretical performance evaluation studies. The contribution of this paper is exposed in section 4. It consists in a performance management information model based on an instrumentation approach. Finally, section 5 draws some conclusions and deals with future works.

2. P2P Networking

Peer-to-peer (P2P) networking is built on a distributed model where peers are software entities which play both the role of client and server. Today, the most popular application domain of this model is file sharing with applications like E-mule, Napster, Gnutella and Kazaa among others[1]. However, the P2P model also covers many additional domains [Oram, 2001] like dis-

[1] www.zeropaid.com

tributed computing (Seti@Home [Anderson, 2001], the Globus Project [Foster and Kesselman, 1999]), collaborative work (Groove[2], Magi[3]) and instant messaging (Jabber[4], JIM [Doyen et al., 2003]). To provide common grounds to all these applications, some middleware infrastructures propose generic frameworks for the development of P2P services (Jxta [Oaks et al., 2002], Anthill [Babaoglu et al., 2002]).

The P2P model enables valuable service usage by aggregating and orchestrating individual shared resources [Milojicic et al., 2002]. The use of existing infrastructures that belong to different owners reduces the costs of maintenance and ownership. The decentralized topology increases fault tolerance by suppressing any central point of failure, and improves both load balancing and scalability. At last, the distributed nature of algorithms and some embedded mechanisms allow participating peers to maintain a great level of anonymity.

2.1 Management of P2P Applications

While some applications use built-in incentives as a minimal self management feature [Mischke and Stiller, 2003], advanced management services are required for enterprise-oriented P2P environments. The latter are the focus of our attention and deserve the availability of a generic management framework.

The first step toward this objective has consisted in designing a generic management information model for P2P networks and services that can be used by any management application as a primary abstraction. The work we have done in this direction has led to a model [Doyen et al., 2004a] that aims at providing a general management information model, that addresses the functional, topological and organizational aspects for such a type of application. We have chosen CIM [Bumpus et al., 2000; Distributed Management Task Force, Inc., 2005] as the framework for the design of our model because of its richness in terms of classes covering several domains of computing that can be easily reused and extended. This way, CIM allows any P2P application to subclass our model in order to provide dedicated classes that will represent the specific application features at best. Instances of these classes will provide a distributed Management Information Base (MIB) that a management application will use to administrate the application.

As shown on Figure 1, the model we have designed covers the various aspects of the P2P domain. First, it deals with the notion of peer and its belonging to one or several communities. A particular association class allows the link of peers together in order to establish a virtual topology. One may note that, according to the context, different criteria can be considered to link two peers;

[2] www.groove.net
[3] www.endtech.com
[4] www.jabber.org

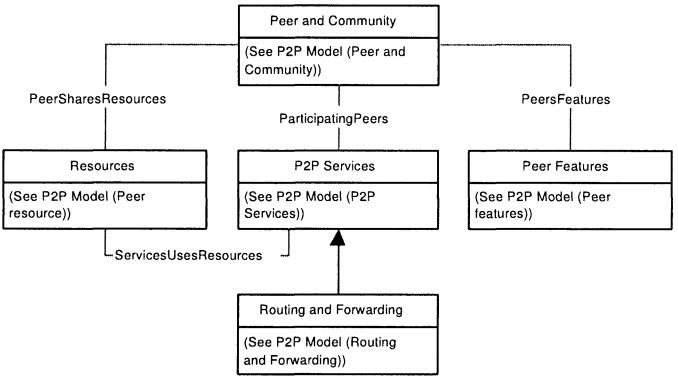

Figure 1. An overview of the CIM extension for P2P networks and services

for example, it can be based on knowledge, routing, interest or technological considerations. Then, our model features the available resources in a community and especially the ones shared by its composing peers. We particularly address the fact that a resource can be spread in a community and thus (1) we differentiate owners and hosts of shared resources and (2) we split resources into physical (e.g. the chunk of a file on a file system) and logical ones (the aggregation of the different chunks). Moreover, the latter are consumed or provided in the context of services that constitutes the fourth aspect of our model. Indeed, a P2P service is a basic functionality that is distributed among a set of participating peers; thus our model enables the global view of a service as well as the local one. Finally, we have identified particular basic services offered by any P2P framework; it concerns, for example, the way packets are routed in a P2P environment that is one of an overlay type. Therefore, we have modeled routing and forwarding services together with the routing tables they generate or use.

In this way, our CIM extension for P2P networks and services provides an abstract view of a P2P network as well as the deployed services located in a manageable environment.

In order to validate our P2P model, we have instantiated it on existing P2P applications. First, we did it on the Chord P2P framework [Doyen et al., 2004b]. Our information model allows the global monitoring of a Chord ring, its participating peers as well as the discovery and stabilization services. Moreover, we have oriented our model toward the performance management and defined metrics dedicated to the particular Chord architecture. The latter concerns the performance of the discovery service, the nodes equity and the ring consistency and dynamics. Stating from this Chord model, we built an abstraction usable for every DHT-based system and we present it in the next section.

3. The Distributed Hash Tables

The principle of a DHT consists in associating a unique key, resulting from a known hash function, to any resource in a P2P community. The collection of all the keys associated to resources represents a hash table that is scattered around nodes by using a common naming scheme for keys and nodes. Finally the use of a particular topology (De Bruijn graphs [De Bruijn, 1946], Plaxton [Plaxton et al., 1997], *d*-torus [Ratnasamy et al., 2001], ...) that exhibits good properties enables an efficient routing of messages. Famous DHTs are Chord [Stoica et al., 2001] that builds a ring topology, CAN [Ratnasamy et al., 2001] that uses a *d*_torus[5] and D2B that is built over De Bruijn graphs, among others. These infrastructures are mainly deployed in large scale data storage [Kubiatowicz et al., 2000; Dabek et al., 2001; Druschel and Rowstron, 2001].

3.1 Existing Performance Evaluations of DHTs

The need of a performance evaluation for DHTs has been established in [Rhea et al., 2003]. Performance evaluation works for P2P architectures and especially DHTs are numerous and of different nature. First, a theoretical model for P2P networks has been established in [Kant, 2003] and the proposal of an analytic model for the performance evaluation of P2P file sharing system is given in [Kant et al., 2002]. Then, a simulation approach has been used in [Tsoumakos and Roussopoulos, 2003] to compare the different P2P search methods. If these works provide a very precise evaluation of DHTs in static cases, their theoretical nature prevents them from dealing with all the phenomena that can appear in case of a real deployment. This is why [Rhea et al., 2003] proposes a benchmarking evaluation approach for Chord [Stoica et al., 2001] and Tapestry [Zhao et al., 2001] with a real implementation under test in the PlanetLab [Peterson et al., 2000] testbed.

Given all these works, the motivation of the design of an information model for the performance evaluation of DHTs are:

- **Integration of a management framework:** In case of a real deployment, performance of DHTs may collapse, due to phenomena of a different nature, like a strong join or leave movement of nodes or the heterogeneousness of the links between peers. This context clearly shows the need for a management approach which can monitor a DHT, evaluate its current performance and act consequently.

- **Standardization:** current works use different metrics for evaluating DHTs which makes the comparison of infrastructures hardly possible.

[5] An circular Euclidean space of dimension *d*

Of course, such a comparison doesn't aim at establishing that a particular DHT is performing better or worse than another one, but it can show differences that can be improved in future development.

4. A performance model for DHTs

In order to provide a generic model that can feature the performance of any DHT, we have identified the aspects common to all the DHTs and deduced a consistent way to evaluate them. As described on Figure 2, the different criteria we have selected are related to the lookup mechanism, the maintenance of routing tables as well as metadata and finally the insertion and removal of nodes in the DHT.

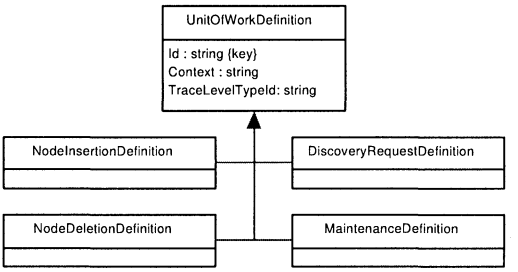

Figure 2. The units of works definitions

4.1 Global Featuring of Distributed Hash Tables

Before characterizing the performance of DHTs, we need to model their components. Figure 3 represents the model we propose to represent, in the management plane, the elements of a DHT. This model is an extension of our general model for P2P networks and services. First, we represent participating nodes through the *DHTNode* class. The *NodeDegree* property represents the number of connections with direct neighbors a node owns. Then, we represent a DHT community (e.g. a chord ring, a CAN torus, ...) with the *DHTcommunity* class. This class provides general information about the grouping of nodes (e.g. *HashMethod*, *NumberOfNodes*, ...) as well as metrics that feature the community behavior (*NodesJoinFrequency*, *KeysMigrationFrequency*, ...). Detailed information about the way we compute these metrics is given in [Doyen et al., 2004b]. Finally, the *PeerResource* represents a resource stored in a DHT. We have chosen to represent it as an abstract class to let the application built over a DHT inherit from it and thus represent any concrete resource.

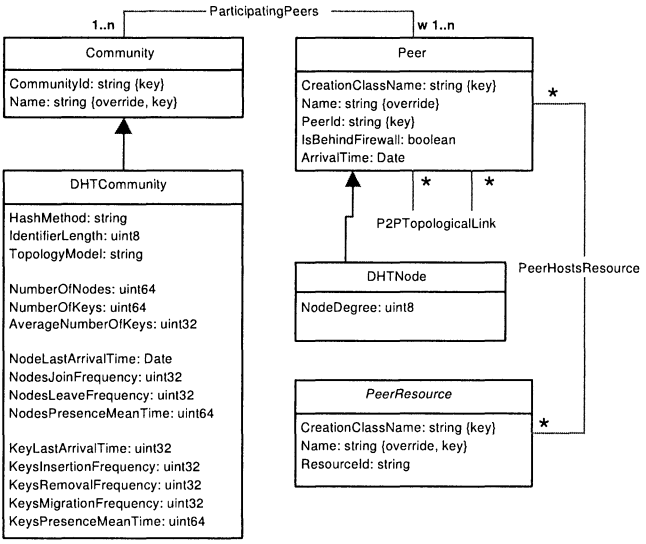

Figure 3. The model for DHT nodes and communities

4.2 The CIM Metrics Model

The CIM metrics model [Distributed Management Task Force, 2003] aims at allowing the management of availability, fault and performance of distributed applications as well as local ones. It provides a set of classes for gathering and managing metrics information. The two concepts introduced in the CIM metric model are the *unit of work* and *metric* ones. According to Figure 4, a unit of work represents a piece of code, currently executed, (a batch job, a network transaction, an end-user command, ...) that has to be characterized. A unit of work can be composed of several sub-units of work that allow different granularity levels. A unit of work is related to a definition that provides informational data about it. Then, the characterization of a unit of work is achieved through metrics. A metric is an assessable criterion that characterizes an aspect of a unit of work. For example, it can be the response time for a web transaction or the amount of used memory for an application. In the same way as the unit of work, metrics are associated to a definition.

4.3 Instrumentation of Discovery Requests

The main goal of DHTs is to allow a user to discover and access resources in a P2P environment. The lookup function provided by the DHT has to remain efficient whatever the number of participants, the network dynamic or the routing distance between resources and requesters. Figure 5 shows the general way

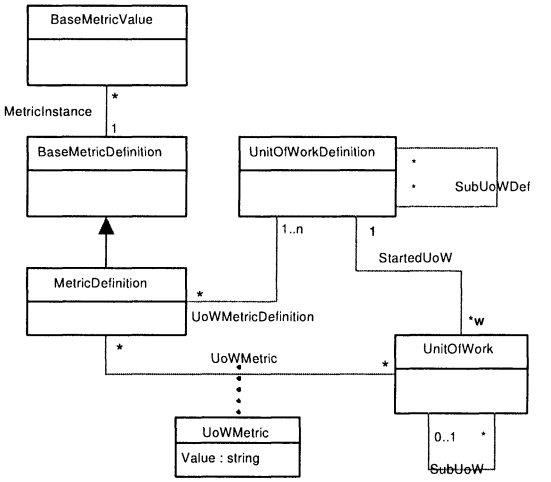

Figure 4. The CIM metrics model

nodes participating to DHTs treat discovery requests. When a request arrives, the receiving node checks in its local cache for the required resource. If the resource is not cached, the node goes through its routing table to find a remote node that can help to satisfy the request. If such a node is found, the request is forwarded. Otherwise, an error message is sent back. This general scenario applies to all existing DHTs.

Based on this general scheme, we have represented the different measurements we propose to feature the discovery service.

The first measurement consists in evaluating the time a node needs to process a request. For a requester node it will represent the time spent to respond to an end-user request and for an intermediate node it will represent to total time spent to process it. Then, we estimated that the local computation time[6] could be interesting to measure. Indeed, in case of too large caches or routing tables size, the amount of time for local processing may be high and such a piece of information is useful to address scalability problems, in particular in environments like the ones described in [Kubiatowicz et al., 2000]. Finally, for all remote calls, we take a snapshot of several parameters in order to evaluate the network load.

The way we model the discovery request operation is shown on Figure 6. As described previously we have distinguished three different units of works. The first one, named *DiscoveryRequest* represents a particular discovery request

[6]it consists in checking the local cache and skim the routing table

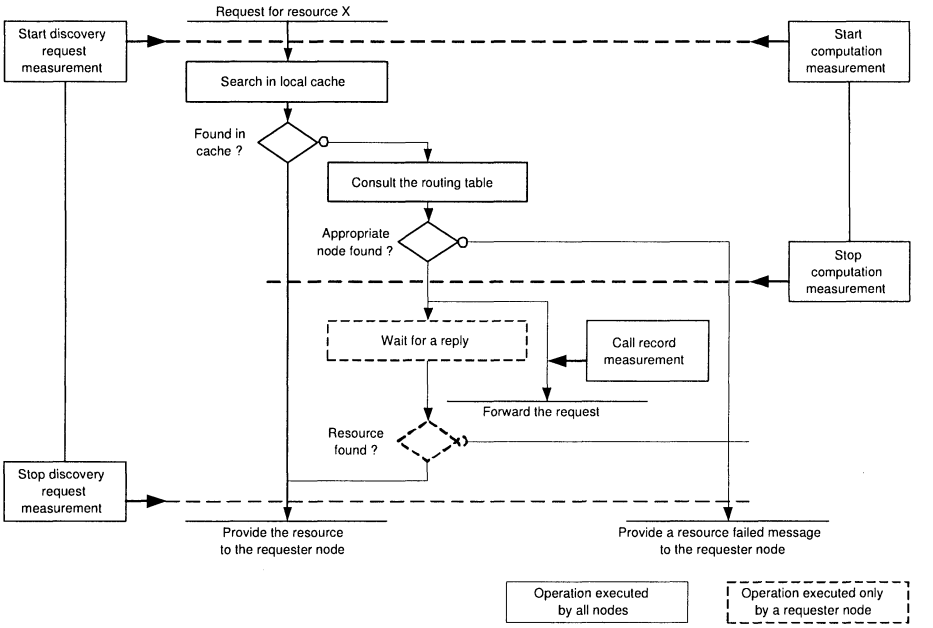

Figure 5. The general scenario for resource discovery

seen by an end user or processed by a forwarding node. The second unit of work is *LocalRequestComputation* and it aims at featuring the local processing operated by nodes. Finally, the *RemoteNodeCall* features the call to a remote node.

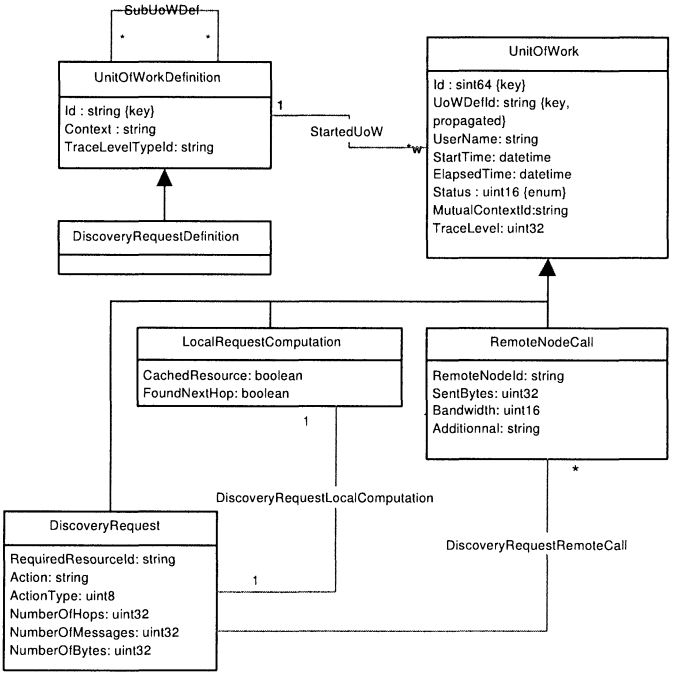

Figure 6. The discovery model

Now, given the preceding context of measurement, we have defined a set of metrics that can feature the discovery requests at best. These are:

- **the average number of hops:** Most existing DHTs theoretically route requests in $O(\log n)$ hops. Nonetheless, in case of strong dynamic of nodes or resources, this value can grow up to $O(n)$ for some of them. This variation needs to be monitored.
- **the global response time:** It corresponds to the time spent to complete a request, from an end-user perspective.
- **the success of the request:** Due to overload or dynamic phenomena, the success rate of discovery requests may collapse; existing resources may become unavailable. This is why we measure the success rate of requests, and consider that this indicator informs about the health of a DHT.

- **the request cost:** Discovery requests have a cost that can be expressed in terms of number of messages, number of bytes and bandwidth that we characterize through this metric.

- **the local computation time:** As described above, we want a node to inform about the time spent processing requests. This time includes both the local cache search and the routing process.

4.4 Performance Evaluation of Other Operations

Existing DHTs exhibit theoretical performances that apply in the context of a stable environment. Nevertheless, in case of a concrete deployment, nodes can come and go in an unpredictable way inducing a dynamic presence and location of resources and routing paths. Such a behavior mandates the presence of a process that can update metadata maintained by nodes so that they will reflect the reality at best. This process is called maintenance process and is executed regularly by all the nodes participating in a DHT.

Moreover, when a node joins a DHT, it becomes responsible for a new set of resources. As a consequence, some metadata have to migrate and involved nodes have to update their routing tables. The same set of changes is done in case of a node departure.

In our performance management oriented information model, we addressed these three operations and we designed dedicated unit of work classes for them. In addition, we have defined standard metrics that feature the global cost of such operations.

4.5 Correlation of the Distributed Measurements

The performance measurements we propose in our model are related to distributed transactions. Indeed, a user request initiated on a particular node may imply the launch of several transactions on different nodes. In this context we need a way to correlate each transaction measurement to a particular context. To do that, we have used the method proposed by CIM in the metrics model. It consists in creating a correlator object that will be attached to each request so that a manager can link it to a particular initial transaction.

Figure 7 displays the correlator class. Among all the properties, the *InitiatorNodeId* provides the identifier of the node that has initiated a transaction; the latter is represented through the *InitiatorTransactionId* attribute. The *previousNode* attribute refers to the last node the transaction has crossed. Finally, the *hopNumber* counts the number of hops crossed by the correlator. This way, we are able to bind each transaction to a context.

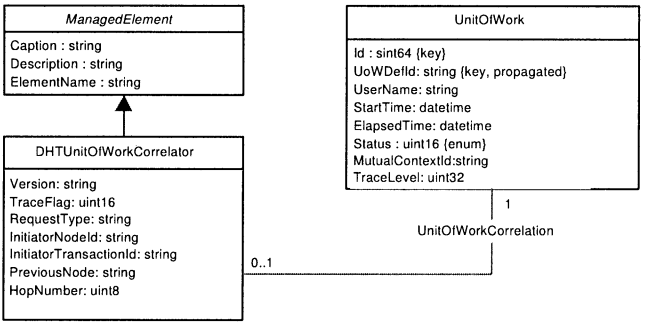

Figure 7. The correlation CIM class

5. Conclusion and Future Work

To propose an efficient alternative to the client/server model, P2P applications need a management framework that ensures service levels without altering the main advantage of these systems: their high distribution. The core of most P2P applications contains a DHT that ensures the discovery and routing aspects with guaranteed theoretical performances. This is why DHTs seem to be unavoidable for the P2P management.

In this paper, we have proposed a performance management information model based on CIM. It enables the monitoring of main DHT operations such as the discovery and the maintenance processes as well as the cost of join and departure of nodes. To do that, we have defined performance metrics and added measurement points within nodes participating to DHTs. This work is generic enough so that any existing DHT implementation can implement it.

Concerning the validation of this model, we plan to achieve it through its deployment. We have instrumented the FreePastry[7] implementation of the Pastry DHT [Rowstron and Druschel, 2001] and Jxta[8]. At this time, all the classes defined in the context of our generic P2P model have been integrated in these two frameworks. The remaining part concerns the performance aspect and we plan to do that by integrating ARM[9] agents into nodes and by exporting collected measurements through a JMX[10] agent.

The work presented in this paper in conjunction with the one presented in [Doyen et al., 2004b] allows us to completely monitor DHTs and to deduce global performance estimators. The current work consists in establishing a

[7] freepastry.rice.edu
[8] www.jxta.org
[9] Application Response Measurement
[10] Java Management eXtension

reactive management behavior from these estimators. Another direction focuses on the management architecture itself and especially the way we could distribute the manager role among managed peers.

References

Anderson, D. (2001). *Peer to peer: Harnessing the Power of Disruptive Technologies*, chapter SETI@Home, pages 67–76. O'Reilly & Associates, Inc.

Babaoglu, O., Meling, H., and Montresor, A. (2002). Anthill: A framework for the development of agent-based peer-to-peer systems. In *Proceedings of the 22th International Conference on Distributed Computing Systems*. IEEE Computer Society.

Bumpus, W., Sweitzer, J. W., Thompson, P., R., Westerinen; A., and Williams, R. C. (2000). *Common Information Model*. Wiley.

Dabek, F., Kaashoek, M.F., Karger, D., Morris, R., and Stoica, I. (2001). Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles - SOSP'01*, pages 202–215. ACM Press.

De Bruijn, N. (1946). *Koninklijke Nerderlanse Academie van Wetenschappen Proc.*, volume 49, chapter A combinatorial problem, pages 758–764. Indagationes Math.

Distributed Management Task Force (2003). Common Information Model (CIM) Metrics Model, version 2.7. www.dmtf.org.

Distributed Management Task Force, Inc. (2005). Common information model v2.7. www.dmtf.org.

Doyen, G., Festor, O., and Nataf, E. (2003). Management of peer-to-peer services applied to instant messaging. In Marshall, A. and Agoulmine, N., editors, *Management of Multimedia Networks and Services*, number 2839 in LNCS, pages 449–461. Springer-Verlag. End-to-End Monitoring Workshop - E2EMON'03.

Doyen, G., Festor, O., and Nataf, E. (2004a). A cim extension for peer-to-peer network and service management. In De Souza, J. and Dini, P., editors, *Proceedings of the 11th International Conference on Telecommunication - ICT'04*, number 3124 in LNCS, pages 801–810. Springer-Verlag.

Doyen, G., Nataf, E., and Festor, O. (2004b). A performance-oriented management information model for the chord p2p framework. In Vicente, J. and Hutchison, D., editors, *Management of Multimedia Networks and Services - MMNS'04*, number 3271 in LNCS, pages 200–212. Springer-Verlag.

Druschel, P. and Rowstron, A. (2001). Past: A large-scale, persistent peer-to-peer storage utility. In *Proceedings of the 8th IEEE workshop on Hot Topics in Operating Systems - HotOS VIII*, pages 75–80. IEEE Computer Society.

Foster, I. and Kesselman, C. (1999). The Globus project: a status report. *Future Generation Computer Systems*, 15(5-6):607–621.

Fraigniaud, P. and Gauron, P. (2003). An overview of the content-addressable network D2B. In *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing - PODC'03*, pages 151–151. ACM Press.

Kant, K. (2003). An analytic model for peer to peer file sharing networks. In *Proc. of International Communications Conference, Anchorage, AL.*

Kant, K., Iyer, R., and Tewari, V. (2002). A performance model for peer to peer file-sharing services. In *Accepted for WWW-11 poster session.*

Kubiatowicz, J., Bindel, D., Chen, Y., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Weimer, W., Wells, C., and Zhao, B. (2000). Oceanstore: An architecture for global-scale persistent storage. *SIGARCH Computer Architecture News*, 28(5):190–201.

Milojicic, D., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins, S., and Xu, Z. (2002). Peer-to-peer computing. Technical Report HPL-2002-57, HP Laboratories.

Mischke, J. and Stiller, B. (2003). Peer-to-peer overlay network management through Agile. In Goldszmidt, G. and Schonwalder, J., editors, *Proceedings of the 8th symposium on Integrated Network Management - IM'03*, pages 337–350. Kluwer Academic Publisher.

Oaks, S., Traversat, B., and Gong, L. (2002). *Jxta in a nutshell*. O'Reilly.

Oram, A., editor (2001). *Peer-to-peer: Harnessing the Power of Disruptive Technologies*. O'Reilly & Associates, Inc.

Peterson, L., A., Anderson;, Culler, D., and Roscoe, T. (2000). Planetlab: A blueprint for introducing disruptive technology into the internet. In *Proceedings of the First ACM Workshop on Hot Topics in Networks (HotNets-I)*, Princeton, NJ.

Plaxton, C. G., Rajaraman, R., and W., Richa A. (1997). Accessing nearby copies of replicated objects in a distributed environment. In *Proceedings the 9th annual ACM Symposium on Parallel Algorithms and Architectures*, pages 311–320. ACM Press.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. (2001). A scalable content addressable network. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures and Protocols for Computer Communication - SIGCOMM'01*, pages 161–172. ACM Press.

Rhea, S., Roscoe, T., and Kubiatowicz, J. (2003). Structured peer to peer overlays need application driven benchmarks. In Kaashoek, F. and Stoica, I., editors, *Proceedings of the 2nd International Peer-to-Peer Systems Workshop (IPTPS'03), Berkeley, CA*, volume 2735 of *LNCS*. Springer-Verlag.

Rowstron, A. and Druschel, P. (2001). Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms - Middleware'01*, number 2218 in LNCS, pages 329–350. Springer-Verlag.

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures and Protocols for Computer Communication - SIGCOMM'01*, pages 149–160. ACM Press.

Tsoumakos, D. and Roussopoulos, N. (2003). A comparison of peer-to-peer search methods. In *Sixth International Workshop on the Web and Databases, San Diego, USA*.

Zhao, B., Kubiatowicz, J., and Joseph, A. (2001). Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division, U. C. Berkeley.