Uenew : A Cross-Browsing Communication System Based on Peer-to-Peer Networks

Ryohei Banno, Haruhiko Sato, Satoshi Oyama, Masahito Kurihara

***Abstract*—In this paper, we propose a communication system that enables users to share experiences of web browsing in real-time. This system groups web users of each web page dynamically and enables them to share comments in each group. In general, high performance servers are required to manage a huge number of groups. In contrast, the proposed system does not need any server to realize large-scale communication system because it uses the Peer-to-Peer technologies. To realize the proposed system, we use Scribe, which is one of the algorithms for application layer multicast. We extended Scribe so that it could treat archives of multicast group. We implemented the extended algorithm as APIs in Java, and evaluated them by some emulation experiments.**

***Index Terms*—peer-to-peer network, overlay network, structured overlay, application leyer multicast, distributed hash table**

I. Introduction

In the real world, we can converse and share experiences with those who are at the same scene by chance. In the web-world, however web users who are visiting the same web page cannot communicate with each other freely. Some web pages have communication systems such as BBS (Bulletin board system), but there are many web pages which have no such systems.

In this paper, we propose a communication system that enables web users to share experiences of web browsing with those who are visiting the same web page. The name of the system is "Uenew", which originates from the word *uenewsar* of Ainu language and means *enjoy speaking together*. Uenew imports a concept of *coming across to somebody* like the real world into the web world.

Uenew groups web users of each web page dynamically and enables them to share comments. For realizing this system, there are two problems. First, in general, for managing a huge number of groups of web pages, we require high performance servers which need high cost to prepare and maintain. Second, too much access to the server can lead the crash of whole system because the server is the single point of failure. In fact, many existing web services have such problems.

To avoid those problems, we designed Uenew to use peer-to-peer technologies, instead of depending on the server. Peer-to-peer network has no single point of failure, and need no cost to prepare servers.

There are three nice points in Uenew.

1) Users can communicate in real-time in each group.
   Uenew enables users to see what the people at the same page are buzzing, by push-based information distribution.
2) Comments are stored in each group.
   Users can share comments on a page which has sparse visitors. That is, users not at the same time, but at the same page, can communicate with each other.
3) Uenew is constructed on Peer-to-Peer network.
   There are no specific servers.

Uenew uses the middleware Overlay Weaver [1] to construct peer-to-peer network. However we can not realize the second point above, archiving comments, by using original Overlay Weaver. So we extended the Overlay Weaver and implemented as APIs, and evaluated the APIs by some emulation experiments.

Manuscript received December 28, 2010; revised February 7, 2011.

The authors are with Graduate School of Information Science and Technology, Hokkaido University, 060-6814, Japan. (e-mail: r_banno@complex.eng.hokudai.ac.jp, haru@complex.eng.hokudai.ac.jp, oyama@ist.hokudai.ac.jp, kurihara@ist.hokudai.ac.jp).

II. Existing services

There are some services which make much of real-time communication like Uenew. A typical service of them is Twitter [2]. However Twitter differs from Uenew at the point of that Twitter provides users with communities based on relations between human and human. Uenew can create new human relations, because it provides users with communities based on relations between human and web pages.

The other feature of Uenew is that users can share the annotated data on web pages. There are some services which have this feature, such as social bookmarking services, Google SideWiki [3] and so on. These services are based on client-server architecture, and they have the problems above-mentioned in section I.

In addition, these services realize the sharing of data with pull-based information distribution. So they can not provides real-time communication like Uenew.

III. Elemental technology

A. Distributed Hash Table

Distributed hash table (DHT) [4], [5], is a class of peer-to-peer system that constructs a hashtable with plural nodes.

In a DHT, nodes and data are assigned with unique ID. Each node covers a range based on the order of nodes and data in the numerical ID space.

Pastry [6] is a typical DHT using plaxton's algorithm [7]. Each node and key is assigned with 128-bit ID, and the ID is thought as a sequence of digits with base 2^b. In lookup procedure, the length of common prefix between the sequences of target ID and local node's ID is extended one digit at a time. These mechanism enables Pastry to provide short latency and small routing table.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

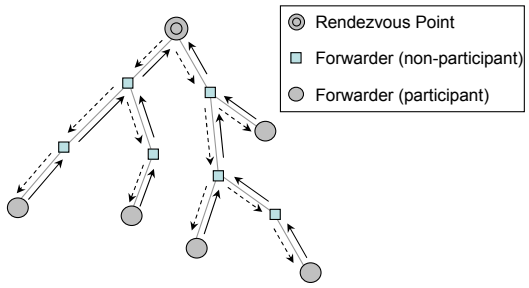

Fig. 1. Multicast tree of Scribe

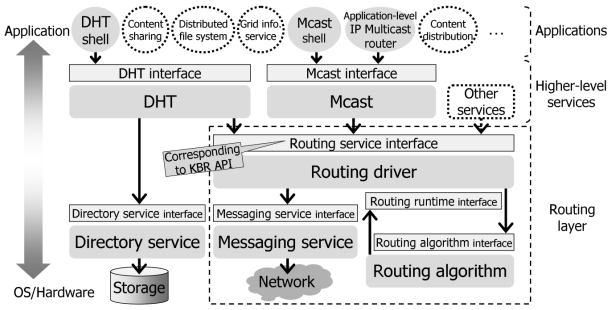

Fig. 2. Components of runtime in Overlay Weaver(Figure 2 from Shudo et al. [1])

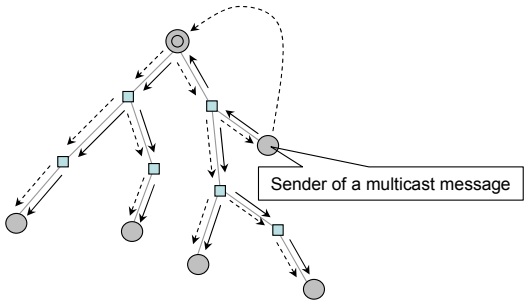

Fig. 3. Difference of multicast routes between Mcast and Scribe

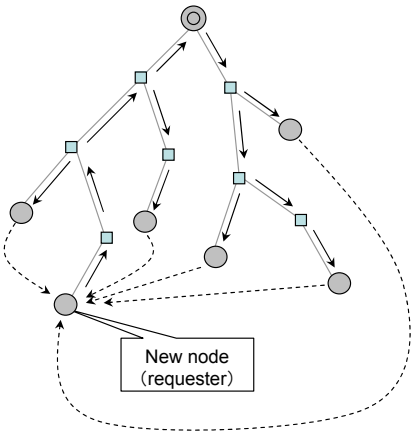

Fig. 4. The way to acquire archives in SAM

B. Application Layer Multicast

Application layer multicast (ALM) is a technology to realize the multicast communication on application layer.

Scribe [8] is one of an algorithms of ALM and uses the Pastry network. On Pastry network, there exists a specific node for each ID. So, the routes from some nodes to an ID constitute a tree structure like Figure 1. The allows of solid line in this figure indicate the routes.

In Scribe, each multicast group has an unique ID and creates its tree. The root node of the tree is called rendezvous point, and the other nodes of the tree are called forwarder.

A node which wants to join a group does lookup with an ID of the group, and is connected to a tree related to the ID. A multicast message is sent to the rendezvous point and transferred toward the children. In Figure 1, the allows of dotted line are the multicast routes.

Because Scribe uses the relationship among nodes on Pastry network, there are two types of forwarders, that is, some forwarders are participating in the group while the others are not. Non-participant nodes only transfer the data from parent node to children.

C. Overlay Weaver

Overlay Weaver [1] is a middleware to construct overlay network. It is based on key-based routing [9] and implemented in Java. Overlay Weaver supports various DHT algorithms such as Chord [4], Kademlia [5], and Pastry [6].

Overlay Weaver is composed of three layers, namely routing layer at the bottom, higher-level services at the middle, and applications at the top. In higher-level services, DHT and Mcast are implemented. Mcast is an implementation of ALM based on Scribe.

IV. Acquiring archives in Scribe

Uenew constructs multicast tree for each URL of web page. Because the original article of Scribe [8] did not refer to the way to treat archives of each group, there is no implementation of Scribe which enables nodes to acquire archives.

Therefore, we extended the Overlay Weaver so that it can provide the function of acquiring archives. We propose two method, Simple Archivable Mcast (SAM) and Advanced Archivable Mcast (AAM).

A. Proposed methods

1) Original ALM of Overlay Weaver: Mcast, which is implemented as higher-level service of Overlay Weaver, provides the function of ALM based on Scribe.

The multicast routes of Mcast are differ from those shown in the original article of Scribe. In Figure 3, the allows of solid line are of Mcast while those of dotted line are of Scribe. In Mcast, a message is transferred not one-way from a rendezvous point but bi-directional from a source node.

2) Simple Archivable Mcast: The most simple way to extend the Mcast so that it can manage archives is to multicast a query of acquiring archives. Nodes which participated in a group must store archives of the group, and a node which want to participate in the group gets archives from existing patricipant. We call this method Simple Archivable Mcast (SAM).

A query of acquiring archives is transferred as the allows of solid line in Figure 4. The allows of dotted line indicate

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

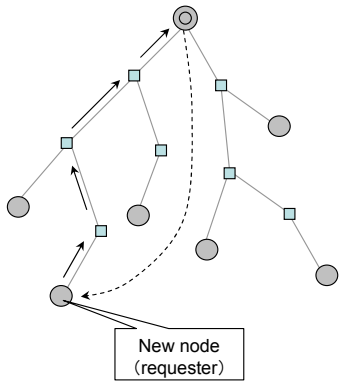

Fig. 5. The way to acquire archives in AAM

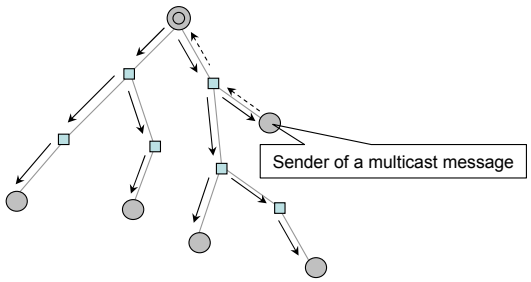

Fig. 6. The way to multicast in AAM

responses to the query. A node which requested archives acquires them from a node which responded first.

The implementation of SAM is relatively easy, but there is a problem of that archives are lost when all participant nodes of the group left.

3) Advanced Archivable Mcast: To avoid the problem of SAM, archives of each group must be stored in somewhere. We propose a new method called Advanced Archivable Mcast (AAM), which is a hybrid of DHT and ALM.

To accumulate/acquire archives, AAM uses DHT network which is constructed as a base of Scribe. Archives of each group is accumulated in the rendezvous point.

The original Overlay Weaver does not suppose that Mcast is used together with DHT. So we constructed a new higher-level service which integrated the functions of DHT and Mcast. Typical procedures of AAM are following:

- Participation in a group
  - Do normal procedure of participation of Mcast.
  - Get archives from rendezvous point by the mechanism of DHT.
- Multicast
  - Put archive to rendezvous point.
  - Rendezvous point transfers the data toward children.

A query of acquiring archives is transferred to rendezvous point along the allows of solid line in Figure 5, and rendezvous point sends archives to the requester (allows of dotted line in the figure). In AAM, archives of a group are available when there is no participant of the group.

Figure 6 indicates the routes of delivering a multicast message in AAM. These resembles to those of Scribe(Figure 3).

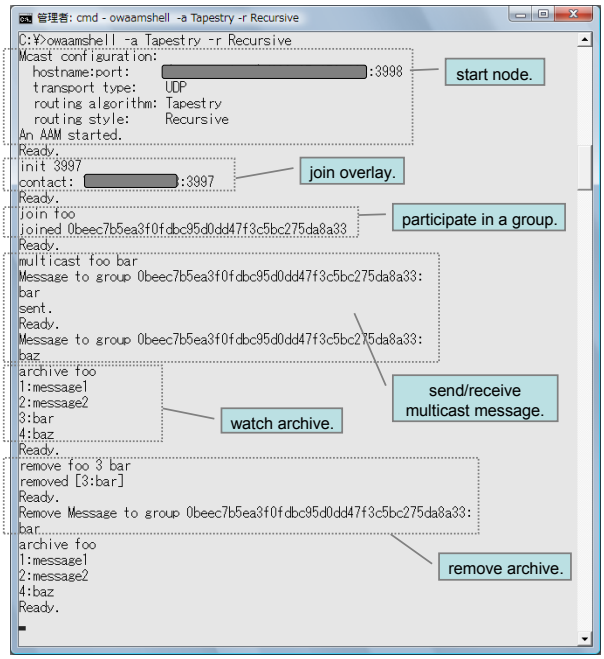

Fig. 7. Execution of AAM shell

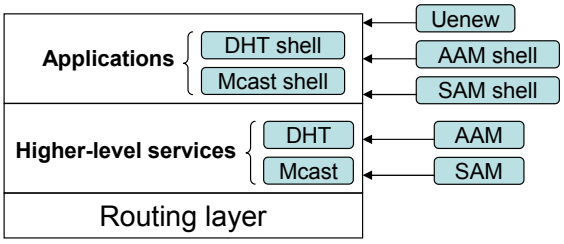

Fig. 8. Additionally implemented modules to Overlay Weaver

B. Implementation as APIs

We implemented SAM and AAM in higher-level services layer of Overlay Weaver. The APIs of the implementation is not specific for Uenew and useful for any system which requires the function of acquiring archives.

In addition, we implemented sample programs called SAM shell and AAM shell in application layer of Overlay Weaver.

These command-line shell enable users to create ALM network and operate nodes easily. Figure 8 shows all modules we implemented to Overlay Weaver.

V. Emulation Experiments

We experimented proposed methods by using an emulator attached to Overlay Weaver. It is possible to run a lot of nodes operated by SAM shell or AAM shell on one machine with the emulator.

In all experiments, we used UDP as transport protocol. We tried five times in each experiment and took the average of them except for the best one and the worst one.

A. Decision of algorithm and style of routing

Overlay Weaver provides some DHT algorithms and two styles of routing. Users of Overlay Weaver can choose one algorithm and one style of routing, and the choice affects the traffic amount on overlay network.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

So we experimented about the traffic to decide algorithm and style at first. As a result, it is found that network has the least amount of traffic when we choose Tapestry algorithm and Recursive routing style. We used this choice in following experiments.

B. comparison of the number of messages

To compare AAM with SAM and Mcast, we experimented about the number of transferred messages by using following emulation scenario.

1) Run 1000 nodes.
2) All nodes join overlay network at 10 msec-intervals.
3) All nodes participate in a group chosen from ten groups randomly at 20 msec-intervals.
4) All nodes multicast a message to the group in which it participated at 20 msec-intervals.
5) All nodes leave from the group at 20 msec-intervals.
6) All nodes participate in a group chosen from ten groups randomly at 20 msec-intervals.

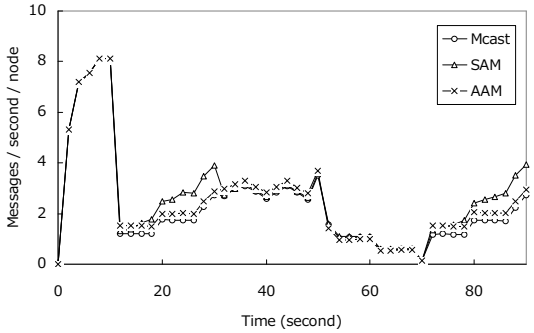

Fig. 9. Comparison of the number of messages among Mcast, SAM and AAM

In Figure 9, the horizontal axis indicates time. And the vertical axis shows the number of messages per node per second. The time window of the number of messages is 2000 msec. There is little difference among Mcast, SAM and AAM during joining overlay (from 0 sec to 10 sec) and leaving groups (from 50 sec to 70 sec). During participation in groups (from 10 sec to 30 sec, and from 70 sec to 90 sec) and multicasting (from 30 sec to 50 sec), There is a clear difference. Mcast is the least, and AAM is somewhat larger than Mcast. SAM is greatly larger than the other two methods.

TABLE I
COMPARISON OF MCAST, SAM AND AAM

	Mcast	SAM	AAM
Acquiring archives	can't	depends	can
efficiency of participating	good	bad	moderate
efficiency of multicasting	good	good	moderate

In Uenew, we can expect that the frequency of participating/leaving groups will be very high, because it will occur every time users move from page to page. So it is a big advantege for Uenew that AAM is more efficient than SAM in participating in groups.

Table I indicates the relative merit of the three methods. In SAM, nodes can acquire archives of a group only when there is some participant nodes in the group. AAM provides moderate efficiency of traffic and the function of acquiring archives which is available at anytime.

C. Evaluation about churn tolerance

In Uenew, the nodes are personal computers of web users. Because it is difficult to predict the behavior of users, we must consider the possibility of that nodes disappear unexpectedly. The frequent occurrence of appearance/disappearance of nodes is called churn.

AAM uses DHT to accumulate/acquire archives, and churn tolerance of DHT has already studied [10]. Therefore we will explain an experiment about churn tolerance of multicasting in AAM.

In Scribe [8], each node which has children sends a small message to children periodically. Nodes can know whether parent node is still alive by the small message. This mechanism realizes high tolerance of churn in Scribe.

Similar mechanism is implemented in Overlay Weaver, which is called GroupRefresher. Each node participating in a group sends a participating query periodically, so the tree of the group is automatically repaired when some nodes disappear.

To see whether AAM has enough tolerance compared to Mcast, we experimented by using following emulation scenario.

1) Run 1000 nodes.
2) All nodes join overlay network at 10 msec-intervals.
3) All nodes participate in a group chosen from 100 groups randomly at 20 msec-intervals.
4) All nodes multicast a message to the group in which it participated at 200 msec-intervals.

Churn occurs while nodes are multicasting.

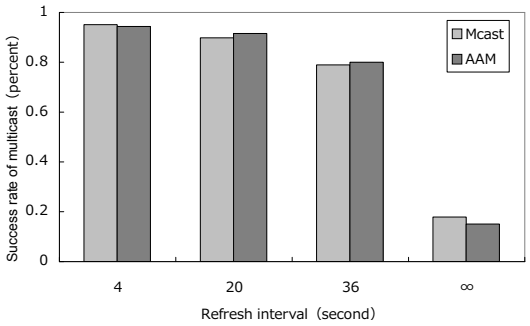

Fig. 10. Comparison of success rate of multicasting between Mcast and AAM

In Figure 10, the horizontal axis indicates interval related to the GroupRefresher. The vertical axis shows the success rate of multicasting. The success rate is the ratio of total frequency of receiving multicasted message of all nodes to the total frequency when there occurred no churn. It is found from Figure 10 that AAM keeps just about the same success rate with Mcast.

VI. IMPLEMENTATION OF UENEW

When a web user is visiting a web page, there is an unique character string called URL. We use the URL as ID of multicast group and implemented Uenew using APIs of

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

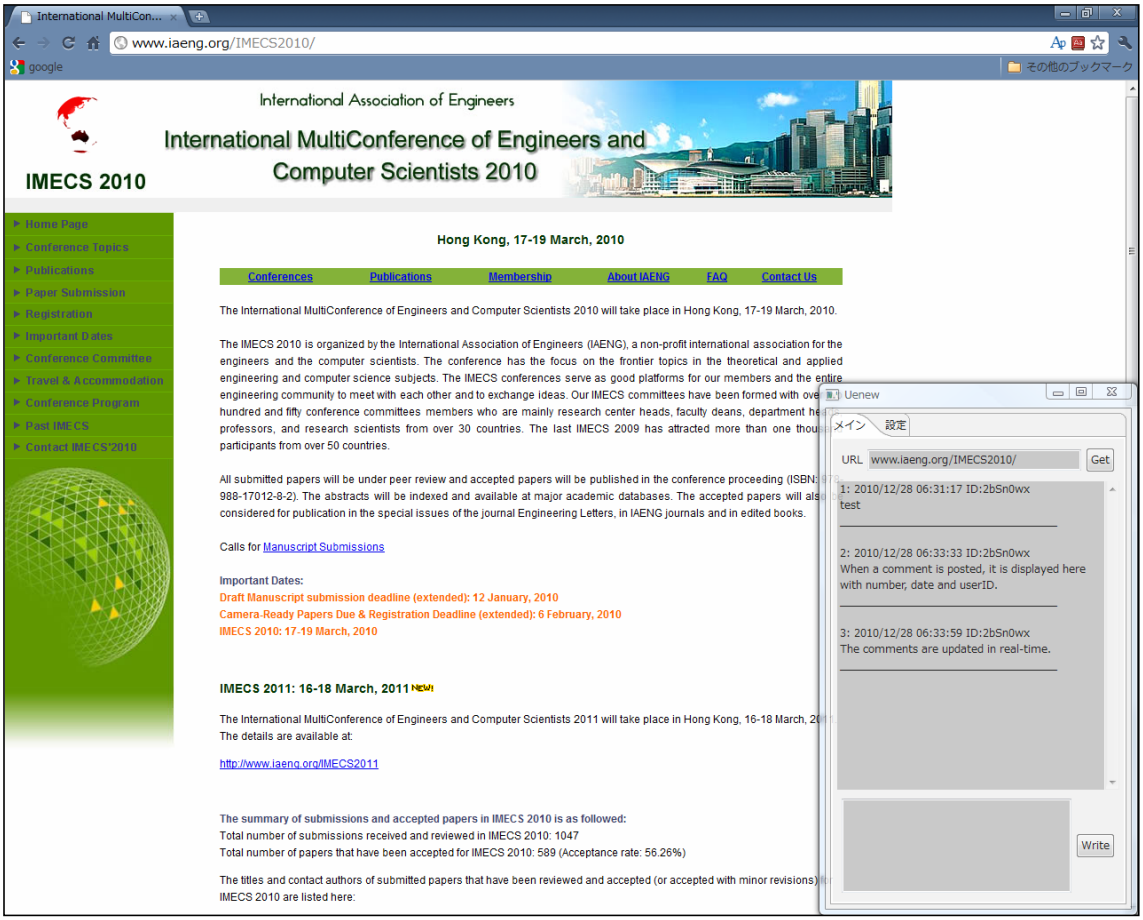

Fig. 11. A screenshot of Uenew

AAM. Uenew get an URL from web browser automatically and dynamically, and participates or creates a group.

For the present, Internet Explorer, Sleipnir and Google Chrome are available for Uenew.

Figure 11 is a screenshot of Uenew. Uenew has two tabs, main tab and setting tab. At the top of main tab, there is an area of URL. Normally the URL in this area is automatically changed, but users also can input an URL manually in this area and create a group related the string.

VII. Conclusions

In this paper, we described the way to realize communication between web users by peer-to-peer network. By using Uenew, it is possible to rendezvous with someone at particular web page or stroll with friends on web space. Uenew requires no server, and never stores something in existing servers which stores web pages. Uenew creates a community for each web page, but the administrator of the web page does not need to do something.

In addition, we proposed new methods to acquire archives of multicast group and implemented them as APIs. The APIs are useful for any system which requires the function of acquiring archives. The APIs and Uenew are available at http://kusshard.complex.eng.hokudai.ac.jp/~r_banno/.

One of the future work of this study is to create communities for each web site, not web page.

References

[1] K. Shudo, Y. Tanaka, and S. Sekiguchi, "Overlay weaver: An overlay construction toolkit," *Computer Communications*, vol. 31, no. 2, pp. 402 – 412, 2008.
[2] Twitter, "Twitter," http://twitter.com/.
[3] Google Inc., "Google sidewiki," http://www.google.com/sidewiki/.
[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *SIGCOMM Comput. Commun. Rev.*, vol. 31, pp. 149–160, 2001.
[5] P. Maymounkov, "Kademlia : A peer-to-peer information system based on the xor metric," *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, 2002.
[6] A. Rowstron, "Pastry : Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," *Lecture Notes in Computer Science*, vol. 2218, pp. 329–350, 2001.
[7] C. G. Plaxton, "Accessing nearby copies of replicated objects in a distributed environment," *Theor. Comput. Syst.*, vol. 32, no. 3, pp. 241–280, 1999.
[8] M. Castro, "Scribe : A large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1489–1499, 2002.
[9] F. Dabek, "Towards a common api for structured peer-to-peer overlays," *Proc. 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, 2003.
[10] K. Shudo, "Churn resilience improvement techniques in an algorithm-neutral dht," *Jounal of Information Processing Society of Japan, Computing System*, vol. 49, no. 2, pp. 1–9, 2008.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)