

Intercontinental Disk-to-Disk Data Transfer Experiment with a Lightweight DTN Software Stack

Purit Phan-udom*, Vasaka Visoottiviseth* and Ryousei Takano†

* Faculty of Information and Communication Technology, Mahidol University, Thailand

purit.pha@student.mahidol.ac.th, vasaka.vis@mahidol.edu

† Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology, Japan
takano-ryousei@aist.go.jp

Abstract—This paper presents a lightweight software stack for data transfer node to carry out massive data transfer, monitor real-time data traffic activity, and provide network infrastructure measurements. It consists of Fast Data Transfer (FDT) as the key application to control all of the data transfers, pairing with OpenTSDB for statistical monitoring in real time and perfSONAR for network statistics. We have demonstrated the proposed DTN software stack by conducting an intercontinental disk-to-disk transfer experiment between Japan and USA. The result shows that the average throughput reaches 6.4 Gbps with a socket buffer tuning.

Index Terms—Data Transfer Node, Big Data Transfer, Disk-to-Disk Data Transfer, High Bandwidth Communication

I. INTRODUCTION

Data-driven science is bringing a paradigm shift to the nature of science. Huge amount of data is generated by instruments on a 24/7 basis, and cyberinfrastructure enables new scientific discoveries and insights from such big data. Science DMZ [1] is a network designed for science data transfers, and the security policy is optimized for high-performance scientific applications rather than for general-purpose computing. It provides a dedicated system for data transfer and network monitoring, i.e., Data Transfer Node (DTN). Some DTN architectures have been proposed, e.g., FIONA and DTN-as-a-Service.

Recently, regional research platforms, including Pacific Research Platform (PRP) [2] and Asia Pacific Research Platform, are emerging. It connects several campus-level Science DMZs through high speed network links for sharing resources with communities. This activity tends to expand at global scale [3].

This paper proposes a lightweight software stack for DTN, specially designed to carry out massive data transfer, monitor realtime data traffic under a variety of conditions, and provide network measurements for the operator. The software stack consists of three major tools: FDT, OpenTSDB, and perfSONAR. The prototype software stack was deployed and tested on the two production computer systems in AIST and the San Diego Supercomputer Center (SDSC). We achieved a stable disk-to-disk data transfer between AIST and SDSC, and

the average throughput was 6.4 Gbps. This result is promising for establishing a global-scale research platform.

This paper is organized as follows. Section II briefly describes the background and related work. Section III presents our proposed lightweight software stack for DTN. Section IV explains the transfer emulation scheme in mini-cluster environment. Next, section V shows the experimental results of intercontinental data transfer between AIST and SDSC. Finally we conclude in Section VI.

II. RELATED WORK

Science DMZ offers a design guideline for handling different aspects of variable factors in the network infrastructure - proper location of devices and connections, systems dedication, performance measurement, and appropriate security [1]. It minimizes the scope of error troubleshooting by isolating the science traffic from the rest of local network. Another benefit is low exposure to security threats.

A. Data Transfer Tools

There are several tools that have been designed and developed for specialized data movement.

SFTP (SSH File Transfer Protocol) and SCP are secure remote file transfer tools. SFTP is the extension of the leading FTP for the assimilation with SSH protocol. It independently works as a subsystem within the security protocol suites. A set of primitive command-line text is used for the client-server communication in a request-response manner for the interaction with each of their data resources.

GridFTP enhances the original FTP as a general-purpose mechanism for secure, reliable, and high-performance data movement [4]. Data transfer can be remotely controlled from a third-party device. Its use of multiple TCP parallel streams improves the bandwidth consumption comparing to a single stream. The protocol also automatically negotiates and adjust TCP buffer sizes, optimizing the performance of data transfer.

Fast Data Transfer (FDT) [5] is an open source application, developed as an add-on module of MonALISA (Monitoring Agent Using a Large Integrated Services Architecture) framework. The application has been written in Java and runs on many platforms with minimal set of dependency libraries required.

B. Software Stack and Services on DTN

Globus Online introduces a hybrid data movement solution, a combination of third-party system and hosted approach, as a hosted SaaS [6]. This can deliver the sophisticated functionality to the users in a way that minimizes their operational cost. The leverage of Amazon Web Service IaaS and GridFTP provides scalability and reliability of the service. Globus Connect was later introduced to resolve the problem when a transfer source or destination is missing GridFTP. The security policy employs the public key authentication protocol where each GridFTP server must have a certificate. Data Channel Security Context (DCSC) was developed afterward to overcome the limitation of traditional Data Channel Authentication (DCAU).

BigData Express [7] is a transfer service middleware that addresses challenges of time-constraint and high-performance in large-scale data transfer applications. It can automatically coordinate the resources based on requirements, task priority and other criteria. Every accepted transfer task is guaranteed not to violate or interfere with each other using an admission control mechanism. Users can gain a quick on-demand summary of the guaranteed end-to-end QoS by the use of SDN-enabled network services. Optimization on multi-DTN systems is aided by implementation of mdtmFTP as its default transfer engine.

PhEDex is a data transfer management system that provides a scalable infrastructure for automatic managing of data maintenance tasks [8]. The system comprises of many agents responsible for different transfer-related tasks; file replication, routing decision, and file pre-staging. It has been designed to utilize any available grid infrastructure synchronously without any constraints in the systems it is embedded into, presenting its adaptability in different network environments.

C. Toward a Smart DTN

To dynamically adjust the computational resources, Z. Liu, et al. [9] proposed a novel approach of designing DTN to be adaptive and respond to observed properties of data flow traffic and the machine status. They applied the concepts of machine learning and knowledge engine that automatically determine how computing resources and data transfer manageability are handled. The knowledge engine generates controlling commands in alignment with current system state data monitored and the learned cost-benefit relationships between

different alternate actions. Many of backend processes of a DTN such as network packet tracing, file transfer scheduling, optimal load balancing, and network congestion control, can then be dynamically adjusted to achieve optimally desirable performance at any given time.

III. LIGHTWEIGHT SOFTWARE STACK FOR DTN

A. Motivation

The satisfactory level of DTN functionality is defined by how well it can serve the task of data transfer between its local storage and a remote site. As its purpose being to specifically handle large bulk data movement on high-speed wide area networks, the specification of both software and hardware must be able to supply high-performance and secure operations that evens with the network capacity, as well as managing of resource or conducting ad-hoc diagnosis on interested factors. The following are operational aspects of software requirements that a conventional DTN should, but not limited to, meet with.

1) *Data Transfer*: The bottom line of a DTN is that it can stably send and receive data at a fast enough rate. GridFTP and its derivation service, such as Globus Online, are commonly employed by the servers in large scientific computing centers. FDT, also a derivation of GridFTP, faces the drawback of TCP transport layer congestion algorithm that narrows the data bandwidth, compensated with buffer tuning and parallel data streams. The other alternative and more simple tools should also be available for relatively smaller transfers - `scp`, `sftp` or `rsync`.

2) *Network Measurement and Testing*: The transferring manager needs to be aware of the network condition and connection availability in order to optimally allocate the computing resources of the DTN. Such information proves crucial to task scheduling and transfer controlling mechanisms. It is highly recommended that a DTN is capable of carrying out regular network testing routine that feeds back the network metrics data for analysis and planning of transfer operations.

3) *Fault Detection and Isolation*: Often the network infrastructure experiences temporary failure or errors. [10] The network-wide anomaly events can hinder and drop data transfer performance. To restore the transfer operation back to normal state, a DTN should be able to quickly and precisely identify the root cause, and effectively troubleshoot the situation.

4) *Auxiliary Operations*: Many supporting tasks are common in the practical work of a DTN. The wide operation range demands a number of individually independent tools responsible for different duties. An occasional view of the system hardware statistics, or monitoring of the data transfer and disk I/O speed, can simplify the degree of complexity in manual control of DTN by the users.

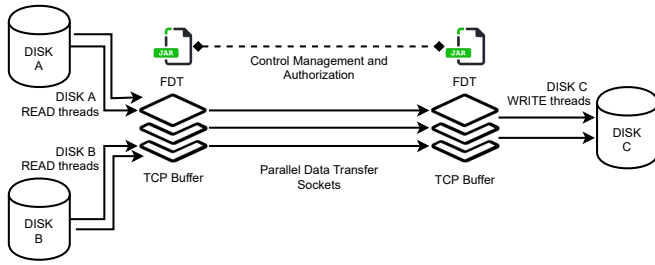


Fig. 1: Fast Data Transfer Functionality Model

5) *Security Risk Mitigation*: To mitigate the security risks of high-value scientific data, a good DTN should not hold in any general-purpose or business productivity software or tools. By having these office applications in the system, the security measure is bound to cover that required for general-purposing computing. An unknown infiltration may hide its presence within the data packets belonging to these software.

B. Software Stack Components

From these areas of requirements, we therefore selected the appropriate set of software and tools to be applied and verified in mini-cluster and ABCI-SDSC testing ground. FDT will be the main facilitator of data movement and traffic. perfSONAR serves the role of network measurement, QoS testing, and fault detection and isolation. OpenTSDB then provides supplementary monitoring of the transfer operation as comprehensible time-series graphs.

1) *Fast Data Transfer*: We used FDT as a file transfer tool between DTNs. FDT is an asynchronous and multi-threaded program. Therefore, it is able to read and write disk-to-disk information over wide area network at a near disk I/O speed. The ability to stream a bulk transfer without network transfer reset between each file forms an advantage for big data transfer applications. By using multiple TCP sockets, the data set is streamed from the sender to the receiver continuously in parallel. Fig. 1 shows the entire data path from the storage of the sender, through the connection link, and to the receiver storage. Two sets of I/O threads for two physically separate disks are used to control the R/W capacity asynchronously from/to the mapped buffer pool. And from the connected streaming sockets, data inside the buffer is carried out of the network interface, through the high-bandwidth WAN in parallel, and headed toward the receiving buffers of the receiver server. Throughout the transfer session, two FDTs periodically communicate and maintain all of the authorization, data transfer management and other controlling mechanisms (on a separate port channel than the data channel).

2) *OpenTSDB*: OpenTSDB [11] is a time-series database server capable of monitoring transfer sessions activities in cooperation with FDT. FDT implements a built-in function module that allows it to associate with OpenTSDB by sending

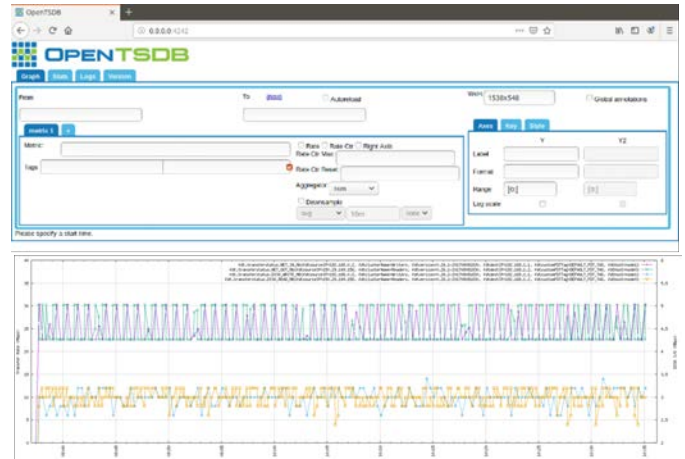


Fig. 2: OpenTSDB dashboard and the plotted query

time series metrics to OpenTSDB server directly. Fig. 2 shows the OpenTSDB dashboard where incoming data points from FDT are transformed into time-domain graphs in real-time manner. The graphs are generated as the result of a graphing query formulated through the query builder (interface above the graphing pane) where it is converted to a gnuplot plotting function. The entire set of data points received belong to more than 10 different metric series, each representing different values associated to the transfer session. Here we visualize 4 series: outgoing data traffic at the sender, incoming data traffic at the receiver, disk read speed at the sender, and disk write speed at the receiver.

3) *perfSONAR*: perfSONAR serves as a supporting network infrastructure tool for testing of network's service quality with different indicating benchmarks [12]. Its underlying principle is the employment third-party command-line `nettest` tools (`iperf3`, `owamp`, `simplestreamer`, etc.) combining with its own set of utility modules.

IV. PRELIMINARY EXPERIMENT ON EMULATED TESTBED

A. Experimental Setting

To emulate real intercontinental big data transfer in our laboratory, we built a mini-cluster testbed environment. It enabled us to test our software stack and tune the performance under several network conditions. The mini-cluster consists of three x86 desktop machines as shown in Fig. 3. Node0 and Node1 were connected through 10 Gigabit Ethernet for big data transfer, and all nodes connected with each other via a Gigabit Ethernet.

Ubuntu 18.04 LTS and our DTN software stack were installed on each node. We used FDT version 0.26.1. The primary purpose of Node2 in fig. 3 is the monitoring of transfer sessions and conducting network metric assessments. This accordingly involved the setup of OpenTSDB server

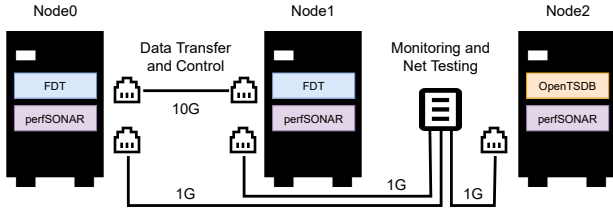


Fig. 3: Mini-cluster experimental environment system

TABLE I: TCP socket buffer tuning setting

Parameter	Value (Byte)
net.core.wmem_max	125000000
net.core.rmem_max	125000000
net.ipv4.tcp_wmem	4096 16384 125000000
net.ipv4.tcp_rmem	4096 87380 125000000

and perfSONAR testpoint module. A perfSONAR bundle was deployed on Node0 and Node1. Certain firewall rules were added in order for the ephemeral ports used by FDT and OpenTSDB to be able to communicate.

We used `netem` emulator to reproduce a similar network condition as in a real ABCI-SDSC link, where the round-trip delay is about 100 milliseconds, and the bottleneck bandwidth is 10 Gbps.

B. Experimental Results

The network latency drops the bandwidth utilization of data transfer over TCP/IP communication. The socket buffer size is a significant performance factor on a high bandwidth-delay product network, and it should be large enough to achieve the theoretical throughput. In this environment as shown in Section IV, the bandwidth delay product was equivalent to 125 MB. Therefore, we set TCP socket buffer `sysctl` parameters as shown in Table I.

To adjust a proper socket buffer size to the network condition, there are two approaches: manual tuning by `setsockopt` system call and automatic kernel tuning. In the former, the socket buffer size is statically set to the given value, where the upper limit is defined by `net.core.rmem_max` and `net.core.wmem_max`. In the latter, on the other hand, the Linux kernel dynamically adjusts the socket buffer size between `net.ipv4.tcp_wmem[0]` and `net.ipv4.tcp_wmem[2]`.

Fig. 4 compares the disk-to-disk transfer performance of two approaches. The manual tuning achieves 3.7 Gbps on average; the automatic tuning achieves 4.5 Gbps on average. Thus, we decided to employ the superior tuning method and apply on the actual ABCI-SDSC experiment.

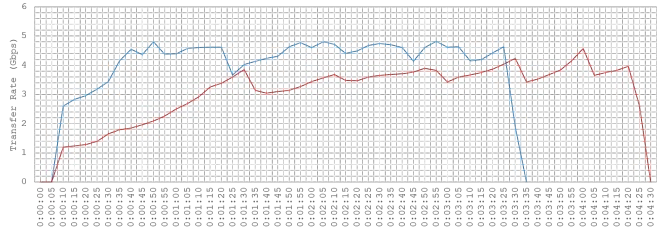


Fig. 4: Manual kernel tuning (red) and automatic kernel tuning (blue) comparison

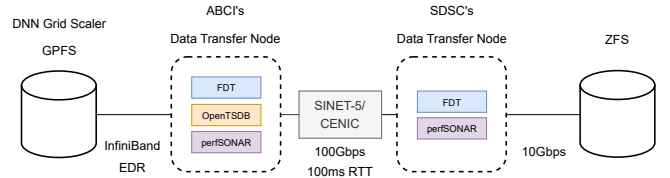


Fig. 5: ABCI - SDSC Software Stack Implementation

V. INTERCONTINENTAL DISK-TO-DISK TRANSFER EXPERIMENT

A. Experimental Setting

We conducted an intercontinental disk-to-disk data transfer between two DTNs on ABCI and SDSC, as illustrated in Fig. 5. The software stack configuration is based on mini-cluster experiments as mentioned above. Table II summarizes the specification of hardware and software in the two systems. Both sites are connected through 100 Gbps backbone networks: SINET5 and CENIC.

AI Bridging Cloud Infrastructure (ABCI) is a large-scale and open infrastructure for open research in the fields of artificial intelligence and big data processing [13]. The authorized limited users only can access DTN for big data transfer. DTN is connected to a 22 PB GRID Scaler (GPFS) through InfiniBand EDR network (100 Gbps). DTN has two 10 Gigabit Ethernet adaptors, and the peak transmission rate is 10 Gbps.

At SDSC, the data transfer node is a CentOS-running server with the access to a ZFS persistent storage unit through 10-Gigabit link. The installed interfaces are two Myricom 10 gigabit ethernet adapters and four 1000Base-T NICs. Its memory capacity is 32GB with one quad-core Intel® Xeon 3.50GHz processor unit.

Table III shows buffered disk I/O throughput with different number of batch jobs and buffer size. This result indicates that disk I/O does not become the bottleneck on this environment if both the buffer size and the parallelism are large enough. Therefore we employ FDT parameters as follows: the buffer size is 1MB; the number of streams is 8.

TABLE II: Specification of DTN on ABCI and SDSC

	ABCI	SDSC
CPU	Xeon E5-2640 v4 @ 2.40GHz x2	Xeon E5-1620 v3 @ 3.50GHz
Memory	DDR4 256 GiB	DDR4 32 GB
NIC	InfiniBand-EDR x2, 10GBASE-T x2, 1000Base-T x2	Myri-10G x2, 1000BASE-T x4
OS	Red Hat Enterprise Linux 7.4	CentOS 7.4.1708
Parallel File System	GPFS	ZFS

TABLE III: Buffered disk I/O performance [MB/sec] by fio benchmark

Parameters		ABCI		SDSC	
buffSize	numStream	Read	Write	Read	Write
4KB	1	1368	950	1820	815
	2	1862	917	2548	1029
	4	843	341	3829	1375
	8	475	284	3278	1545
	16	602	326	4681	1435
	32	744	277	7448	1411
1MB	1	5650	2070	4998	7450
	2	10300	4212	9777	14367
	4	21700	8418	18833	19567
	8	41600	16800	21367	11567
	16	69133	29300	15400	11333
	32	62100	25700	9805	10966

B. Dataset

We prepared a data set of raw zebrafish microscopy imaging data to be used as the transfer payload. It consists of 2977 files and each file size is 193 MB. In this experiment, therefore, we have transferred 574 GB in total.

C. Experimental Result

Using the superior automatic kernel tuning strategy and socket buffer parameters as in Table I, we ran the test scenario between ABCI and SDSC using the data set previously mentioned. The result was an even higher rate of data communication comparing to what was achieved in the mini-cluster. On average, FDT carried out bandwidth utilization at 6.5 Gbps (as shown in Fig. 6), 1.4 times greater than that of the transfer between Node0 and Node1 emulation. We are surprised at this niche margin in the performance level. Our logical explanation suggests this as a result of faster disks I/O speed and internal data transmission acceleration due to better hardware specification.

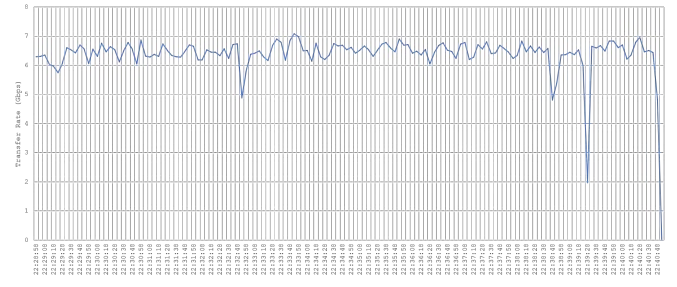


Fig. 6: Disk-to-disk data transfer between ABCI and SDSC

VI. CONCLUSION AND FUTURE WORKS

In summary, we verified the completeness and functionality of a hypothetical software stack for a lightweight DTN in a contained mini-cluster hardware environment. Consisting of FDT, OpenTDSB and perfSONAR, every unit was able to carry out its designated tasks per their capabilities. Next, we replicated an identical implementation on both ABCI and SDSC servers as a two-connecting DTNs. From the usability test case, the data transfer operation returned a desirable performance in terms of reliability and speed. Implementation of OpenTDSB and perfSONAR module is to be delivered in the future period to complete the stack build.

A number of potential future research has been brought up in later discussion. Suggestions for developing of FDT are to support a broadcasting data distribution function and to perform automatic tuning of TCP socket buffer sizes based on the bandwidth-delay product of each network. A comprehensive study of data transmission path could lead to a valid explanation for the occurrence of ABCI-SDSC's superior outcome than mini-cluster output. In this age of ever-increasing tools and services for massive data movement, a comparative analysis with appropriate metrics will lay a firm foundation to the standard and classification criteria of software in this field.

ACKNOWLEDGEMENT

This research work was done during an internship at AIST. This work was supported by the ICT International Collaboration Fund from AIST and Faculty of Information and Communication Technology, Mahidol University. The authors would like to thank the members of Cyber Physical Cloud Research Group for their invaluable comments. We also thank Nadya Williams for helping our experiment at the San Diego Supercomputer Center, University of California San Diego.

REFERENCES

- [1] E. Dart, L. Rotman, B. Tierney, M. Hester, and J. Zurawski, "The science dmz: A network design pattern for data-intensive science," in *SC '13: Proceedings*

of the International Conference on High Performance Computing, Networking, Storage and Analysis, Nov. 2013, pp. 1–10.

- [2] *Pacific research platform*. [Online]. Available: <http://pacificresearchplatform.org/>.
- [3] *The global research platform workshop*. [Online]. Available: <http://grp-workshop-2019.ucsd.edu/>.
- [4] W. Allcock, J. Bresnahan, R. Kettimuthu, and M. Link, “The globus striped gridftp framework and server,” in *SC '05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, Nov. 2005, pp. 54–54.
- [5] *Fast data transfer*. [Online]. Available: <https://fast-data-transfer.github.io/>.
- [6] B. Allen, J. Bresnahan, L. Childers, I. T. Foster, G. Kandaswamy, R. Kettimuthu, J. Kordas, M. Link, S. Martin, K. Pickett, and S. Tuecke, “Globus online : Radical simplification of data movement via saas,” 2011.
- [7] Q. Lu, L. Zhang, S. Sasidharan, W. Wu, P. DeMar, C. Guok, J. Macauley, I. Monga, S. Yu, J. H. Chen, J. Mambretti, J. Kim, S. Noh, X. Yang, T. Lehman, and G. Liu, “Bigdata express: Toward schedulable, predictable, and high-performance data transfer,” in *2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS)*, Nov. 2018, pp. 75–84.
- [8] J. Rehn, T. Barrass, D. Bonacorsi, J. Hernandez, I. Semeniouk, L. A. Tuura, and Y. Wu, “Phedex high-throughput data transfer management system,” 2006.
- [9] Z. Liu, R. Kettimuthu, I. Foster, and P. Beckman, “Towards a smart data transfer node,” in *4th International Workshop on Innovating the Network for Data Intensive Science*, 2018.
- [10] Y. Zhang, S. Debroy, and P. Calyam, “Network-wide anomaly event detection and diagnosis with perfsonar,” *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 666–680, Sep. 2016.
- [11] *Documentation for opentsdb 2.3*. [Online]. Available: <http://opentsdb.net/docs/build/html/index.html>.
- [12] *User guide for perfsonar 4.2.0*. [Online]. Available: <http://docs.perfsonar.net/>.
- [13] *ABCI*. [Online]. Available: <https://abci.ai/>.



Purit Phan-udom (M19) has become a Member of IEEE in 2019. He was born in Bangkok, Thailand on September 1, 1996. He is currently a senior student (4th year), majoring in database and intelligent system, at the Faculty of Information and Communication Technology, Mahidol University, Nakhon Pathom Province, Thailand. He is expected to graduate with a Bachelor degree in computer science in the

year 2020.

He had worked at Japans National Institute of Advanced Industrial Science and Technology (AIST) as an intern researcher, and produced the work published in this conference. Currently,

he is working on a thesis research that aims to develop an automatic tool for analysis of Pythonic idiom usage in open-source git projects. His current research interests concerns in-network computing and software-defined networking.

Mr. Phan-udom is also a member of Faculty of ICTs Software Engineering Research Unit (SERU).



Vasaka Visoottiviseth is an associate professor at Mahidol University, Thailand. She received her Ph.D. degree in computer engineering from Nara Institute of Science and Technology (NAIST), Japan in 2003, M.E. and B.E. degree from Tokyo University of Agriculture and Technology (TUAT), Japan in 1999 and 1997, respectively. Her current research interests are

mobile and wireless computing, Internet traffic measurement, and network security.



Ryousei Takano (M07) received his Ph.D. in Engineering from the Tokyo University of Agriculture and Technology, Tokyo, Japan, in 2008. He became the member of IEEE in 2007. His research interests include operating systems and distributed parallel computing.

He is a research group leader of the Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. He

joined AXE, Inc. in 2003 and then, in 2008, moved to AIST.

Dr. Takano is also a member of IPSJ and IEICE.