# Steering Committees

Steering Committees for ICoICT 2016 :

**Muhammad Ashari**, Telkom University, Indonesia
**Heroe Wijanto**, Telkom University, Indonesia
**Djusnimar Zultilisna**, Telkom University, Indonesia
**Maman Abdurohman**, Telkom University, Indonesia
**Adiwijaya**, Telkom University, Indonesia
**Ari M. Barmawi**, IEEETelkom University, Indonesia
**Muhammad Ary Murti**, IEEETelkom University, Indonesia
**Mike Wald**, IEEEUniversity of Southampton, UK
**Roberto Montemanni**, IEEEUniversity of Applied Sciences of Southern Switzerland, Switzerland
**Lau Siong Hoe**, IEEEMultimedia University, Malaysia

# Program Committees

Program Committees for ICoICT 2016 :

**Dade Nurjanah (TPC Chair)**, Telkom University, Indonesia
**Aniliza Asnawi (Track Chair)**, International Islamic University,Malaysia
**Ari M. Barmawi (Track Chair), Telkom University, Indonesia**
**Athitaya Nitchot (Track Chair)**, Prince of Sonkhla University, Thailand
**Dana S. Kusumo (Track Chair)**, Telkom University, Indonesia
**Emil Kaburuan (Track Chair)**, SICS, Sweden
**Jimmy Tirtawangsa (Track Chair)**, Telkom University, Indonesia
**Moch Arif Bijaksana (Track Chair)**, Telkom University, Indonesia
**Gunawan Putu Harry (Track Chair)**, Computational Sciences, School of Computing, Telkom University, Indonesia
**Yanti Rusmawati (Track Chair)**, Telkom University, Indonesia
**A Adiwijaya**, Telkom University, Indonesia
**Abdelhak Zoubir**, Darmstadt University of Technology, Germany
**Aede Musta'amal**, Universiti Teknologi Malaysia, Malaysia
**Agus Arifin**, Institut Teknologi Sepuluh Nopember, Indonesia
**Anna Antonyová**, University of Prešov in Prešov, Slovakia
**Anto Satriyo Nugroho, BPPT, Indonesia**
**Anton Satria Prabuwono**, King Abdulaziz University, Saudi Arabia
**Areeb Alowisheq, King Saud University, Saudi**
**Aries Susanty**, Universitas Diponegoro, Indonesia
**Arvind Patil**, Centre for Distance Engineering Education Programme, IIT Bombay, India
**Asep Najmurrokhman**, University of Jenderal Achmad Yani, Indonesia
**Ashok Kumar Veerasamy**, University of Turku, Finland

**Bambang Hidayat**, Institut Teknologi Telkom, Indonesia
**Bayu Hendradjaya**, Institut Teknologi Bandung, Indonesia
**Bernardi Pranggono**, Sheffield Hallam University, United Kingdom
**Bong Jun Choi**, The State University of New York (SUNY) Korea, Korea
**Chan Basaruddin**, Universitas Indonesia, Indonesia
**Chau Yuen**, Singapore University of Technology and Design, Singapore
**Chih-Wei Liu**, National Chiao Tung University, Taiwan
**Chun Yeow Yeoh**, Telekom Reseach & Development Sdn. Bhd., Malaysia
**Claudiu Mihaila**, Stratified Medical, United Kingdom
**David Rydeheard**, The University of Manchester, United Kingdom
**Deni Saepudin**, Institut Teknologi Telkom, Indonesia
**Dina Shona Laila**, University of Southampton, UK
**Dwi Hendratmo, Bandung Insitute of Technology, Indonesia**
**Edi Kurniawan**, Indonesian Institute of Sciences, Indonesia
**Eduard Babulak**, The Institute of Technology and Business in Ceske Budejovice, Czech Republic
**Emir Mauludi Husni**, Institute of Technology at Bandung, Indonesia
**Endang Chumaidiyah**, Telkom Institute of Technology, Indonesia
**Eniman Syamsuddin**, Institut Teknologi Bandung, Indonesia
**Farhana Sarker**, University of Liberal Arts Bangladesh, Bangladesh
**Florian Buether**, University of Lübeck, Germany
**Giles Reger**, The University of Manchester, United Kingdom
**Go Hasegawa**, Osaka University, Japan
**Goh Kah Ong Michael**, Multimedia University, Malaysia
**Gunawan Ariyanto**, Universitas Muhammadiyah Surakarta, Indonesia
**Hafizi Muhamad Ali**, Yanbu University College, Saudi Arabia
**Haidi Ibrahim**, Universiti Sains Malaysia, Malaysia
**Hans Dulimarta**, Grand Valley State Universuty, United Kingdom

**Harco Leslie Hendric Spits Warnars**, Bina Nusantara University, Indonesia
**Harry Santoso**, Universitas Indonesia, Indonesia
**Hasan Tinmaz**, Middle East Technical University, Turkey
**Hassan Chizari**, Universiti Teknologi Malaysia, Malaysia
**Heroe Wijanto**, Telkom University, Indonesia
**Hertog Nugroho**, Bandung State of Polytechnic, Indonesia
**Hilman Pardede**, Indonesian Institute of Sciences, Indonesia
**Hisar Maruli Manurung**, University of Edinburgh, United Kingdom
**Hsiang-Cheh Huang**, National University of Kaohsiung, Taiwan
**Hyun-Ho Choi**, Hankyong National University, Korea
**I Putu Agung Bayupati**, Udayana University, Indonesia
**Indrawati Indrawati**, Telkom University, Indonesia
**Iwan Syarif**, Politeknik Elektronika Negeri Surabaya (PENS), Indonesia
**Jarutas Pattanaphanchai**, Prince of Songkla University, Thailand
**Johanna Octavia**, Parahyangan Catholic University, Indonesia
**Kewalin Angkananon**, Suratthani Rajabhat University, Thailand
**Khairayu Badron**, International Islamic University Malaysia, Malaysia
**Khoirul Anwar**, Japan Advanced Institute of Science and Technology, Japan
**Kiki Adhinugraha**, Monash University, Australia
**Kyoko Ito**, University of Kyoto, Japan
**Kyung Hyune Rhee**, Pukyong National University, Korea
**La-or Kovavisaruch**, National Electronics and Computer Technology Center, Thailand
**Lau Siong Hoe**, Multimedia University, Malaysia
**Leony Lidya**, University of Pasundan, Indonesia
**M. Udin Harun Al Rasyid**, Politeknik Elektronika Negeri Surabaya (PENS) – Indonesia, Indonesia
**Maman Abdurohman**, Telkom University, Indonesia
**Marc Stelzner**, University of Lübeck, Germany
**Masayu Leylia Khodra**, Institut Teknologi Bandung, Indonesia
**Maulahikmah Galinium**, Swiss German University, Indonesia
**Mewati Ayub**, Maranatha Christian University, Indonesia
**Mettina Veenstra**, Saxion University, The Netherlands
**Micke Rusmerryani**, Research Lab AP Channel, Indonesia
**Mike Wald**, University of Southampton, United Kingdom
**Mochamad Ashari**, Institut Teknologi Sepuluh Nopember (ITS) – Surabaya, Indonesia
**Mohd Helmy Abd Wahab**, Universiti Tun Hussein Onn Malaysia, Malaysia
**Mohd Zool Hilmie Mohamed Sawal**, Universiti Teknologi MARA Kedah, Malaysia
**Muhamad Sadry Seman**, International Islamic University Malaysia, Malaysia
**Nakarin Pinprathomrat**, RMUTT, Thailand
**Nargis Bibi**, Fatima Jinnah Women University, Pakistan
**Nattiya Kanhabua**, Aalborg University, Denmark
**Niruwan Turnbull**, Mahasarakham University, Thailand
**Noor Zaman**, King Faisal University KSA, Saudi Arabia
**Norhidayah Azman**, Universiti Sains Islam Malaysia (USIM), Malaysia
**Nurazzah Abd Rahman**, Universiti Teknologi MARA, Malaysia
**Nurul Fariza Zulkurnain**, International Islamic University Malaysia, Malaysia
**Odiel Estrada Molina**, University of Informatics Science, Cuba
**Oliver Kleine**, University of Luebeck, Germany
**Ong Thian Song**, Multimedia University, Malaysia
**Onjira Sitthisak**, Thaksin University, Thailand
**Parman Sukarno**, Politeknik Pos Indonesia, Indonesia
**Philip Moore**, Lanzhou University, P.R. China
**Prachya Boonkwan**, National Electronics and Computer Technology Center, Thailand
**Preecha Tangworakitthaworn**, Mahidol University, Thailand

**Ririn Agustin**, Universitas Pasundan, Indonesia
**Roberto Montemanni**, Dalle Molle Institute for Artificial Intelligence (USI/SUPSI), Switzerland
**Rosaura Palma-Orozco**, Instituto Politécnico Nacional, Mexico
**S. Agrawal**, Delhi Technological University (DTU) Formerly Delhi College of Engineering (DCE), India
**Saiful Akbar**, Institut Teknologi Bandung, Indonesia
**Sanjay Singh**, Manipal Institute of Technology, India
**Satria Mandala**, UniversitiTeknologi Malaysia, Malaysia
**Sebastian Ebers**, University of Lübeck, Germany
**SeungHwan Won**, University of Southampton Malaysia Campus, Malaysia
**Shaufiah Shaufiah**, Telkom University Indonesia, Indonesia
**Shuhaida Shuhidan**, Universiti Teknologi MARA, Malaysia
**Shuhaili Talib**, International Islamic University Malaysia, Malaysia
**Shukor Sanim Mohd Fauzi**, Universiti Teknologi Mara, Perlis Campus, Malaysia
**Sian Lun Lau**, Sunway University, Malaysia
**Siong Hoe Lau**, Multimedia University, Malaysia
**Siva Dorairaj**, Software Education, New Zealand
**Soo King Lim**, Universiti Tunku Abdul Rahman, Malaysia
**Suronapee Phoomvuthisarn**, Mahanakorn University of Technology, Thailand
**Takashi Kurimoto**, NII, Japan
**Tasanawan Soonklang**, Silpakorn University, Thailand
**Thomas Basuki**, UNPAR, Bandung, Indonesia
**Tien Kusumasari**, Telkom University, Indonesia
**Udayangi Perera**, University of Auckland, New Zealand
**Vania Estrela**, Universidade Federal Fluminense, Brazil
**Wan Fatinhamamah Wan Ahmad**, Universiti Putra Malaysia, Malaysia
**Watsawee Sansrimahachai**, University of the Thai Chamber of Commerce, Thailand
**Yahaya Coulibaly**, Universiti Teknologi Malaisia, Malaysia
**Yan Liu**, ASDC Illinois, Singapore
**Yasin Kabalci**, Nigde University, Turkey
**Yati Rohayati**, Institut Teknologi Telkom, Indonesia
**Yixin Wang**, Institute for Infocomm Research, Singapore
**Yu Yuan-Chih**, National Taipei University of Technology;Chinese Culture University, Taiwan
**Zoya Pourmirza**, University of Newcastle, United Kingdom

# Organizing Committees

Organizing Committees for ICoICT 2016 :

**Yanti Rusmawati (The Conference Chair)**, Telkom University, Indonesia
**Tjokorda Agung B. W. (The Conference Co-Chair)**, Telkom University, Indonesia
**Sri Widowati (Finance Chair and Treasurer)**, Telkom University, Indonesia
**Mediana Mayang Kencana (Finance Chair and Treasurer)**, Telkom University, Indonesia
**Dade Nurjanah (Technical Program Committee)**, Telkom University, Indonesia
**Dana S. Kusumo (Technical Program Committee)**, Telkom University, Indonesia
**Jimmy Tirtawangsa (Technical Program Committee)**, Telkom University, Indonesia
**Gia Septiana (Secretariat Chair)**, Telkom University, Indonesia
**Rita Rismala**,Telkom University, Indonesia
**Elen T. Sepiati** , Telkom University, Indonesia
**Aniq Atiqi**, Telkom University, Indonesia
**Said Al Faraby (CFP, Marketing, Sponsor Chair)**, Telkom University, Indonesia
**Anton Herutomo**, Telkom University, Indonesia
**Annisa Aditsania**, Telkom University, Indonesia
**Kurniawan Ramadhani**, Telkom University, Indonesia
**Mira Kania S.(Logistic Chair)**, Telkom University, Indonesia
**Anditya Arifianto**, Telkom University, Indonesia
**Sahrul Mubarok**, Telkom University, Indonesia
**Nungki Selviandro (Event Chair)**, Telkom University, Indonesia
**Sidik Prabowo**, Telkom University, Indonesia
**Danang Junaedi**, Telkom University, Indonesia
**Rimba Whidiana C. (Publications Chair)**, Telkom University, Indonesia
**Moch Arif Bijaksana (Publications Chair)**, Telkom University, Indonesia
**Fazmah A. Yulianto (Publication Chair)**, Telkom University, Indonesia
**Agung Toto Wibowo (Publications Chair)**, Telkom University, Indonesia
**Dawam Dwi Jatmiko Suwawi (Publications Chair)**, Telkom University, Indonesia

# KNOPPIX : Parallel Computer Design and Results Comparison Speed Analysis Used AMDAHL Theory

Yudhi Arta[1], Evizal Abdul Kadir[2], Des Suryani[3]
[1,2,3]Teknik Informatika, Fakultas Teknik
Universitas Islam Riau, Pekanbaru,
Riau, Indonesia 28284

yudhiarta@eng.uir.ac.id, evizal@eng.uir.ac.id, des.suryani@eng.uir.ac.id

***Abstract* — Parallel computing namely the unification of multiple computers or servers into a single unit that can work simultaneously or processes simultaneously. Parallel computing is creating programs and processes run faster as more and more CPU used. Basically parallel computing using network media, but that is characteristic in particular is how to resolve the issue. Problems encountered here is how me-rendering an image that looks speedup PC and serial value contained by parallel computing.**

***Keywords — Parallel Computing, Server, CPU***

## I. INTRODUCTION

Parallel processing is the use of more than one CPU to execute a program in simultaneously. Ideally, parallel processing makes the program run faster as more and more CPU used. But in practice, it is often difficult to divide a program that can be executed by the CPU different without regard of them.

In recent years, power management has become one of the most important issues in the server cluster. Some of the proposed methods on a single server can be expanded to a server cluster. In this case there is the same way to implement DV / FS and reconfiguration of the cluster, using a threshold value, based on the utilization of system load to keep the processor frequency as low as possible, the nodes are less active and extend the scheme of feedback control for cluster [1].

Parallel computing is one of the most interesting technologies are important since the invention of electronic computers in the 1940s. Breakthroughs in parallel processing is always evolving and getting a place in addition to other technologies since the Age of Awakening (1950), Era Mainframe (1960), Era Minis (1970), Era PC (1980), and the Age of Computers Parallel (1990s). With a wide range of influences on the development of other technologies, and how technology is changing the perception of the computer, it is understandable how important the parallel computing. The essence of parallel computing is hardware, software, and applications.

Parallel processing is information processing closer to the average of the manipulation of data elements to one or more of the settlement process of a problem. Therefore, parallel computing can be interpreted that a computer with many processors are able to perform parallel processing.

## II. LITERATURE REVIEW

Parallel computing is a computing technique simultaneously by utilizing multiple independent computers simultaneously. Typically used for data processing with the capacity very large (industrial environment, bioinformatics and others) or because of the demands of computing that much. Parallel computer is a single computer with multiple internal processors or multiple computers that are connected by an Interconnection Network (IN) parallel computers can be grouped, based on the organization of memory, into two basic architecture that is: the system shared memory and system memory are spread (distributed memory). In a distributed memory system using inter-process communication mechanism for the exchange of messages (message passing). A parallel program with the exchange of messages can be written using a high level programming language (such as Fortran, C / C ++), and include (or call) a message exchange library MPI (Message Passing Interface) or PVM (Parallel Virtual Machine). In this research used MPI library. There are two parallel program structures: Single Program Multiple Data (SPMD) and Multiple Program Multiple Data (MPMD). In SPMD structure, there is only one source program and each processor will execute a copy of the program. While the structure of the MPMD, there are several sources of programs and each processor execution in different programs [2-5].

Parallel programming is a computer programming technique that allows the execution of commands or operations simultaneously in both the computer with a single processor or many processors with parallel machines CPU. The main purpose of parallel programming is to improve computing performance. The more things can be done simultaneously (in the same time), the more work can be completed. Parallel computing requires such algorithms, programming languages and compilers.

### *A. PVM*

PVM is software that makes your computer into a bunch of looks like a large virtual computer system. A set of computers that will be involved in the problem solving

ISBN: 978-1-4673-9879-4 (c) 2016 IEEE

process should be defined in advance, in order to perform its functions. The computers involved in computing could be homogeneous, with the same platform, or heterogeneous, with different platforms, the origin of them can communicate with each other. PVM can handle all shipping processes, conversion of data and task scheduling.

PVM system is composed of two parts, the first part is a daemon named PVMD. PVMD activated on each computer that will form a virtual machine. The second part is the routine library PVM interface that contains a collection of primitive commands to operate the processes of the library. Library is routinely used in parallel application programs written in the C programming language, C ++, or FORTRAN 77. The application in the programming language C and C ++ libraries are connected through libpvm3.lib, whereas the application in the programming language FORTRAN 77 can take routines in libfpvm3. lib [6].

PVM provides the facility to create a number of processes that do not depend on the number of processors. Each process is identified using a code (task ID) and mapped to the processor automatically, or it can also be set by the programmer. PVM programs are generally governed by the master-slave models, which is a process that is executed first became master and turn on the entire client by calling PVM_Spawn. The routine will automatically run all processes in the system PVM. Another way to run the process is to call a routine that returns a code PWM_Mytid task ID of the process. Before the exit of the PVM system, all client processes must be switched off from PVM by calling the routine PWM_Exit.

Communication between processes in the system message passing PVM carried out using PVM routine commands like PWM_Send and PWM_Recv. All the routine delivery of messages is done asynchronously, while all routine receipt of the message can be either synchronous or asynchronous. There are three stages in sending a message in PVM, namely:

1. Initialize buffer with regular delivery PWM_Initsend
2. Pack the contents of the message into the buffer by calling the routine PWM_pk *. The data can be of type byte packed, complex, double, float, integer, and character. The types declared by replacing the "*" and give the appropriate type for each parameter in the routine pvm_pk *. For example, the data to be packed type float, then use pvm_pkfloat.
3. Send the message to the destination processor by calling PWM_send or PWM.cast process that receives the message must open the message packet in the receiving buffer in accordance with the format of message delivery [7].

PVM also provide routine PVM setup to set options in the PVM system, such as printing an error message automatically, the level of debugging (debugging level), and the method of setting the lines of communication. The most common example of the use of PWM_setopt is allowing direct communication between the task in PVM. PWM_setup (PVMR) route, PvmRouteDirect). With this command automatic communication bandwidth in the network will be duplicated

*B. MPI*

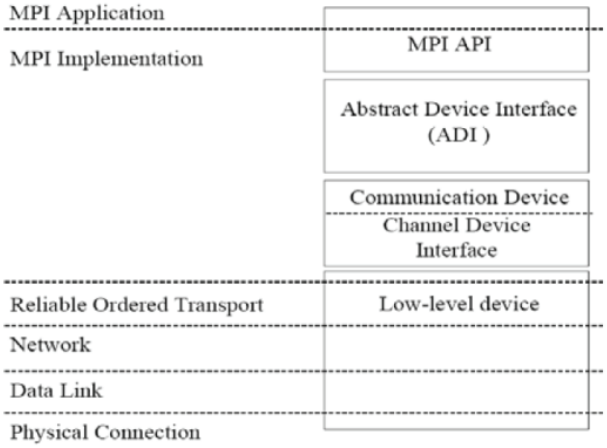
According to [6], in the implementation using the MPI library functions that can be called from the program C, C ++, or Fortran. Almost the same as PVM, MPI is also portable to a variety of architectures. One of the latest implementation at the time of writing this paper that runs in the Windows environment is MPICH. Judging from the application side, MPI can only be used with the model of the single program multiple data (SPMD), while the PVM can be used with SPMD models and multiple program multiple data (MPMD). SPMD model is physically indicated by program master and slave are fused, while MPMD program is indicated by a separate master and slave, so the slave can undertake the task vary from one node to another node.

MPI is a programming technique that is based on data parallel with Single Program Multiple Data (SPMD). The point is that each process executes the same program but using different data. For sharing data, a process explicitly transmits data to the receiver which also receives data explicitly. MPI is not a new programming language, but MPI is a subprogram library that can be called from C and Fortran program 77. There are two implementations of MPI for the grid: GridMPI made by the National Institute of Advanced Industrial Science and Technology (AIST) and MPICH- G2 created by Argonne National La boratory (ANL). GridMPI is the implementation of MPI for the distance between computers up to 500 miles connected to a network speed of 1 to 10 Gbps while MPICH-G2 is MPI implementations using the Globus Toolkit (grid computing middleware standards) in a WAN environment [8].

Abstraction application program created by the user declared in the layer API. Library functions are available on the MPI stated in the header mpi.h. MPI activation starts by running the command MPI_Init (& argc, & argv); the main program, followed by determining the ranking of each node that runs an application program with command MPI_Comm_rank (MPI_COMM_WORLD, & my_rank); my_rank is a positive integer, zero means the program is running on the master computer, otherwise worth is not equal to zero means the program is running on the slave computer. MPI_COMM_WORLD is a constant that has been defined to control the processes that exist at the time of MPI starts. To determine the number of processors (node) that actively uses the command MPI_Comm_size (MPI_COMM_WORLD, & p) [9].

Communication is done sequentially from the top layer to the physical layer; on the receiver side opposite applies,

that is, from the physical layer to the top. For example there MPI_Send command (message, strlen (message) +1, MPI_CHAR, dest, tag, MPI_COMM_WORLD); message (message) along strlen (message) +1 and the type of character (MPI_CHAR) is sent to the destination processor (dest). At ADI layer of the message received by Send_handle namely controlling data transmission in the lining of the ADI, then the Device Channel layer packet is received by the controller MPID_SendControl and MPID_SendChannel. Furthermore, the low-level layers of the communication device carried by the protocol available. For example MPICH runs on Windows 2000, the protocol used is TCP/IP. On the receiver side, the communication layer or channel controller device interface has MPID_ControlMsgAvail and MPID_RecvAnyControl. Both of these controllers forward the packet to a layer of ADI. At this layer, there are two controllers, namely: PostedRecv_Handles and UnexpectedRecv_Handles, each of which is used to determine the delivery of packets transmitted with right and wrong. Furthermore, at the application layer, the data is received by the command MPI_Recv (message, len, MPI_CHAR, source, tag, MPI_COMM_WORLD, & status) and to terminate the MPI used MPI_Finalize command. Figure 1 shows architecture of parallel computer [10-12].




Fig, 1. Architecture of parallel computer.

## III. Reserach Methodology

In the methodology of this research it can be concluded that the problems faced are not optimal resource usage. The methodology can be used to increase the availability of a server is to create a load balancing cluster. With the server cluster is able to overcome the problems encountered over dependence faced by her agency itself. Given these resource it can be built a cluster that can help performance in terms of infrastructure and others. The cluster itself also works to increase performance and also stability workings of the system itself. Job descriptions will be explained in the framework that will explain the procedures and steps - steps that AKN faced in building this server cluster. The initial stage is to analyze and design what - what is needed in the construction of these server clusters. Then precede with the test results together with a report on the performance of the cluster in a table [13-14]. Figure 2 shows a research framework in this study.

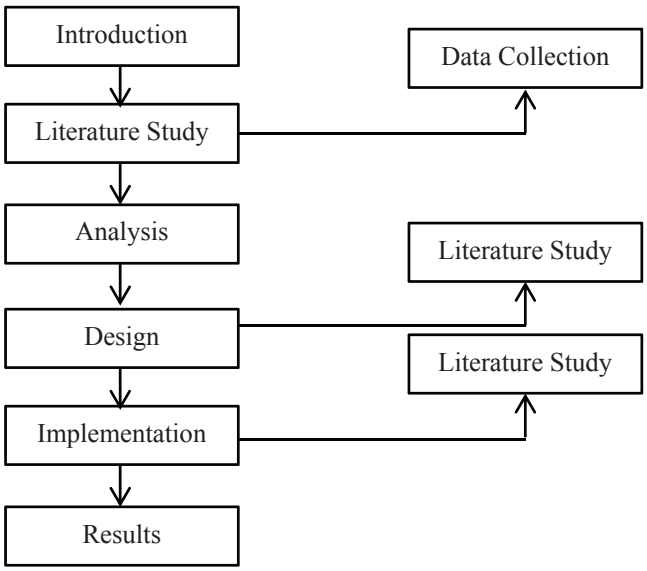



Fig. 2. Research framework.

## IV. Results and Discussion

In the discussion section, the authors will display any related artifacts of every process and activity undertaken at each phase of the methodology according to the DAD. Each phase has a goal process activity, where the activity of these processes and produce artifacts or applications built documentation. As mentioned above, DAD divide the stages of software development becomes inception, construction and transition.

### 4.1 Topology design

For cluster topology design is almost the same as the topology of grid computing. In this case, the cluster has one master and several nodes / client / slave which will be managed by the master cluster. Almost all large companies have implemented since the beginning of their computing cluster built.

In the figure 3 shows is used by enterprise-scale institutions to manage the webserver that is the heart of their investment in the competition in cyberspace. To build a parallel computing for corporate, usually there are vendor who are willing to wake or provide these facilities at considerable cost relatively high. Some vendors include IBM, Microsoft, Dell, and other major vendors. Figure 3 also shows that can be explained that the purpose of the cluster is to back up the webserver rapid access to funds to help node in front of him, so that the database and file sharing that is used can be accessed and assisted by rapid cluster.

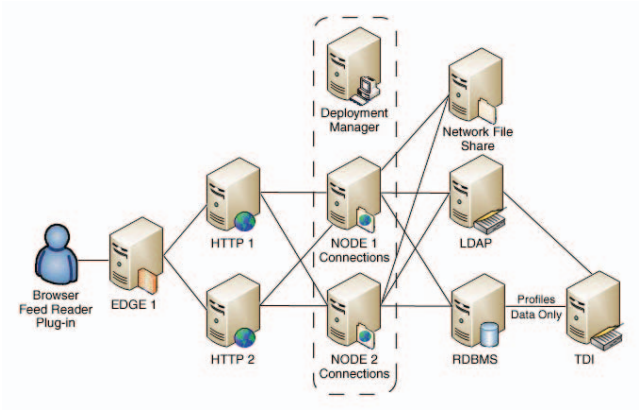



Fig. 3. Topology load balancing for enterprise.

The rendering of the results we've been getting the value of the rendering and the allocation of resources that we use when rendering. The results can be seen in figure 4.

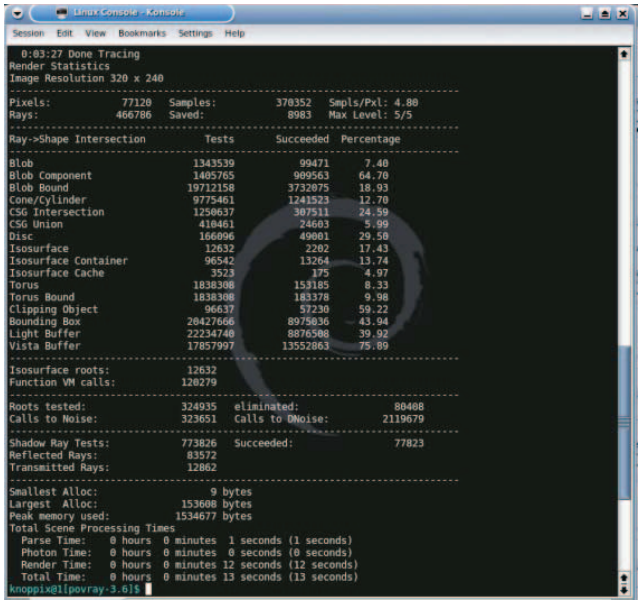



Fig. 4. Rendering timing result.

Some test results then we will make conclusions and we will measure the value Amdahl's law to find how much the value of the series contained in the CPU which cannot be contested. For the results of the rendering, we can deduce the cluster results into Table 1 proficiency level:

## 4.2. Amdahl Law

Having obtained the results of the speedup we have used above, then we need to determine the value of the series that does not affect the parallel processing. Serial values are useful determine what percentage of the process that are not affected from parallel processing results in terms of both hardware and software. As for Amdahl's Law is:

$$Sp \leq 1 \frac{1}{f + \frac{1-f}{2}} \leq \frac{1}{f}$$

*Sp* = *SpeedUp* yang didapat dengan P buah procesor
f = Bagian dari program yang harus dieksekusi secara serial (0..1)

Hukum Amdahl untuk 2 *node* :

$$Sp \leq 1 \frac{1}{f + \frac{1-f}{2}} \leq \frac{1}{f}$$

$$1{,}17 \leq 1 \frac{1}{f + \frac{1-f}{2}}$$

1,17 f + (1,17/2) [1-f] = 1
1,17 f + 0,58 – 0,58f = 1
0,58 f = 1 – 0,58
0,58 f = 0,42
f = 0,42 / 0,58
f = 0,72 dijadikan persen :
f = 0,72 *100%
f = 72 %

TABLE 1. RENDERING RESULTS WITH 2 NODES

| Pengujian Cluster | Parse Time (Sec) | Photon Time (Sec) | Render Time (Sec) | Detik (2 Komputer) (sec) |
|---|---|---|---|---|
| 1 | 1 | 0 | 11 | 12 |
| 2 | 1 | 0 | 11 | 12 |
| 3 | 1 | 0 | 11 | 12 |
| 4 | 1 | 0 | 11 | 12 |
| 5 | 1 | 0 | 11 | 12 |
| 6 | 1 | 0 | 11 | 12 |
| 7 | 1 | 0 | 11 | 12 |
| 8 | 1 | 0 | 11 | 12 |
| 9 | 1 | 0 | 11 | 12 |
| 10 | 1 | 0 | 11 | 12 |
| 11 | 1 | 0 | 11 | 12 |
| 12 | 1 | 0 | 11 | 12 |
| 13 | 1 | 0 | 11 | 12 |
| 14 | 1 | 0 | 11 | 12 |
| 15 | 1 | 0 | 11 | 12 |
| | | | Rata-Rata | 12 |

## V. CONCLUSION

The entire cluster can be run in accordance with the desired and the results of the rendering speedup how we get the data needed for parallel processing or clustering. The results of testing 2 or 3 node exists, we have got what percentage of the value contained in the serial parallel processing though not so valid because of limitations. With the value of the series, we can conclude there is still a considerable value for parallel processing that cannot be contested. The values of this series are located in two layers, namely: software and hardware. Value serialized in software, there are some instructions that are not affected by the process of clustering or parallel processing. Instruction or algorithm that is applied in parallel processing is inviolable. Also in the operating system it in clusters or parallel processing does have limitations to instruct the

command is executed. The value serialized in hardware can be regarded as limitations Ethernet card, cable used and also some other factors. In order to get the fraksial more significant or more valid, it can be proved by building a cluster with 10 nodes or additional multiple nodes again. To prove the serial process that cannot be contested can we prove it by opening the cluster coding process which is considered the series and can be further analyze.

## REFERENCES


1. Xinying Zheng and Yu Cai, "Achieving Energy Proportionality In Server Clusters", International Journal Of Computer Networks (IJCN), Volume II, No. 23, 2007.
2. Seminar Kudang B, et al, "Uji Dan Aplikasi Komputasi Paralel Pada Jaringan Syaraf Probabilistik (PNN) Untuk Proses Klasifikasi Mutu Tomat", Jurnal Teknologi, 34, 2006.
3. Chokchai Leangsuksun, et al, "Highly Reliable Linux HPC : Self Awareness Approach", 2000.
4. John Shalf, "The New Landscape of Parallel Computer Architecture", Conference Series 78, No.1, 2007.
5. Anton Dwi Laksono, A.Benny Mutiara MQN and Brahmantyo Heruseto, "Analisis Perbandingan Antara Cluster Openmosix Dengan Mpi Terhadap Aplikasi Rendering Pov-Ray", 593, 2004.
6. Heru Suhartanto, "Kajian Perangkat Bantu Komputasi Parallel Pada Jaringan PC", Volume 10, Number 2, 2006, pp.72.
7. Andria Arisal, Wiwin Suwarningsih, and Nuryani, "IGNITER : Membangun Komputer Cluster Dengan Cepat", Volume IV, Number 2, 2010, pp.95.
8. Said Mirza Pahlevi, "Komputasi Grid Dan Paralel", volume 15, 2008.
9. Dewi Frestyani Maya Putri, Singgih Jatmiko and Benny Mutiara, "Perbandingan Kinerja Cluster OpenMosix dengan Disk dan Tanpa Disk", 2010.
10. Fani Fatullah, A.Benny Mutiara MQN and Chandra Yulianto, "Analisa Kinerja Cluster Dengan Pustaka Mpich Terhadap Perkalian Matriks", 2004.
11. Pandriya Sistha and Ade Jamal, "Membangun PC Cluster Berbasis Interkoneksi Statis Menggunakan Topologi Ring dan Torus", 2006.
12. Prema S Desai and Deven M.Thakur, "International Journal of Trends In Engineering, Clustering Of Virtual Disk Design Of Clustered Servers For Storage Management", volume 1, Number 2, 2009, pp. 267.
13. Tittel, Ed., "Schaum's Outline: Computer Networking", Erlangga, 2004.
14. Wahana Komputer, "Administrasi Jaringan dengan Linux Ubuntu 11", Andi, 2011.