Distributed Dynamic Failure Detection

Ahmad Shukri Mohd Noor
[1]Department of Computer Science, Faculty of Science and Technology,
Universiti Malaysia Terengganu, 21030, Malaysia
ashukri@umt.edu.my

Mustafa Mat Deris[2], Md Yazid Md Saman[1] and Emma Ahmad Sirajudin[1]
[2]Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, 86400, Johor, Malaysia
Email: mmustafa@uthm.edu.my, yazid@umt.edu.my, emma.ahmads@gmail.com

***Abstract*—Failure monitoring and detection phase is a critical part in providing a scalability, reliability and high availability in current distributed environment. Heartbeat style of interaction is a widely used technique. This technique is utilized for detecting a fault where it monitors the heartbeats of system resources continuously in a very short interval. However, this approach has its limitations as it requires a period of time to detect the faulty node, causing delay in the impending recovery procedures. This paper presents a fault detection mechanism and service using hybrid heartbeat mechanism and dynamic estimated time of arrival (ETA) for each heartbeat message. This technique introduces the use of index server for indexing the transaction and operates dynamic hybrid heartbeat mechanism and pinging procedure for fault detection. The evaluation outcome signifies the use of the hybrid heartbeat mechanism in reducing approximately 30% of the time taken to detect faults compared to existing techniques and provides a basis for a customizable recovery action to take place.**

***Index Terms*—failure detection, fault-tolerance,d istributed systems.**

I. Introduction

An effective and efficient Fault tolerant methodology is central in developing reliable dependable systems. Consequently, fault tolerant computing has become an active research area [1][2] and fault detection is the first essential phase for developing any fault tolerance mechanism or fault tolerant systems[3]. Fault detection is the most important aspect of a fault-tolerant distributed system and thus is the standard necessity to achieve scalability, dependability and high availability in distributed environment/systems.

Fault detection identifies and provides information on the faults of the components of these systems[4]. Based on the system design and the assumptions about fault characteristics of components, the study found that fault-detection latencies covered from 55% to 80% of non-functional periods.

This non-functional phase occurs when a system is unaware of a failure (failure detection latency) and periods when a system attempts to recover from a failure (failure-recovery latency)[5][14].

Even though the development of fault detection mechanism in large scale distributed system is subject to active research it still has some lacks[4]. Matthew Gillen *et al* introduced Scalable, Adaptive, Time-Bounded Node Failure Detection (NFDS)[6]. Since network traffic, CPU load becomes unpredictable. However the paper[6] did not discuss the algorithm on how to tailor the detectiontime in case of network congestion of CPU overload. The paper only demonstrated the mathematical and experimental comparison across detection time, mistake rate, network bandwidth, and message loss rate.

One of the most popular fault detector service in grid computing is the Globus Heartbeat Monitor (GHM)[7]. GHM provides a fault detection service for applications developed with the Globus toolkit. It was developed under the assumption that both the grid generic server and the heartbeat monitor run reliably[8]. However, few bottlenecks have been identified; GHM scales badly in the number of members that are being monitored[9], requiring developers to implement fault tolerance at the application level which is difficult to implement and does have high-overhead[10], Failure Detection and Recovery Services (FDS)[11], improves the Globus HBM with early detection of failures in applications, grid middleware and grid resources. The FDS also introduces an efficient and low-overhead multi-layered distributed failure detection service that spare grid users and developers the burden of grid fault detection and fault recovery. However, this technique has a weakness as it requires a period of time before detecting a fault and consequently delays the recovery actions. The problem arises due to the un-indexed status of each transaction and the need to wait for a certain time period.

K. C. W. So, and E. G. proposed latency and band-width-minimizing for failure detector and proposed a generic way to detect faulty nodes but exclude flexibility (i.e., ability to support different types of applications)[12]. L. Falai and A. Bondavalli presented an experimental evaluation of different estimations and safety margins of a distributed push failure detector [13].

Corresponding Email address: ashukri@umt.edu.my

doi:10.4304/jsw.9.5.1342-1347

In this paper, we propose a dynamic failure detection mechanism by integrating dynamic heartbeat detection mechanism with pinging service. In this technique, all the message transactions must be indexed for references and for further action to be taken during recovery process. It is named hybrid due to the integration of pinging to the heartbeat mechanism. We focus on the problem of detecting fail-stop crash failures; a failure in which a crash results in the component to transit permanently to a state that allows other components to presume that it has aborted (e.g., by ceasing to send periodic \i-am-alive" messages). This research focuses on improving failure detection methodology for a large scale distributed environment. This work also aims at reducing detection time, eliminating false alarm warning, scalability and providing the system with more accurate information for recovery and fault tolerance purposes.

The rest of the paper is organized as follows: Section 2 presents the proposed failure detection mechanism architecture, illustrates the component functionalities to detect the failure as well as failure detection workflow. The comparison with existing technique is demonstrated in Section 3. Section 4 discusses the result analysis. Finally, in Section 5 we conclude the paper.

II. Dynamic Framework

This section elaborates the dynamic heartbeat framework (DHF) as illustrated in Figure 1. The DHF comprises three main components that are Node Manager (NM), Index Server (IS) and Heartbeat Monitor (HBM).

i) Node Manager (NM)

In order to facilitate the DHF, firstly each member node needs to register itself for inventory at NM. NM provides the means to register a member node to the Index Server by storing and updating the required information to be monitored. Once a node is registered on NM, the information provided by the node will be added and indexed in the Index Manager (IM). The IM will receive notification when the node is added to the index or changed since it was last indexed.

ii) Index Manager (IM)

The IM will be notified each time the NM registers new node or updates the node information since last indexed. This information is automatically updated in IM. During monitoring process, the IM receives a log file generated by the HBM for each heartbeat message

iii) Heartbeat Monitor (HBM)

HBM is responsible for monitoring the state of the registered nodes. In the case of contradiction from their usual behaviors, it will notify the IM so necessary actions can be taken. Each node periodically sends a message indicating its aliveness to the HBM. Then, HBM generates a status to be sent to the Index Server to update the current status of the node. If the HBM does not receive a heartbeat within certain duration HB_{max}, the particular node is rendered failed. Accordingly, the IM will provide sufficient information for recovery process after pinging has taken place.

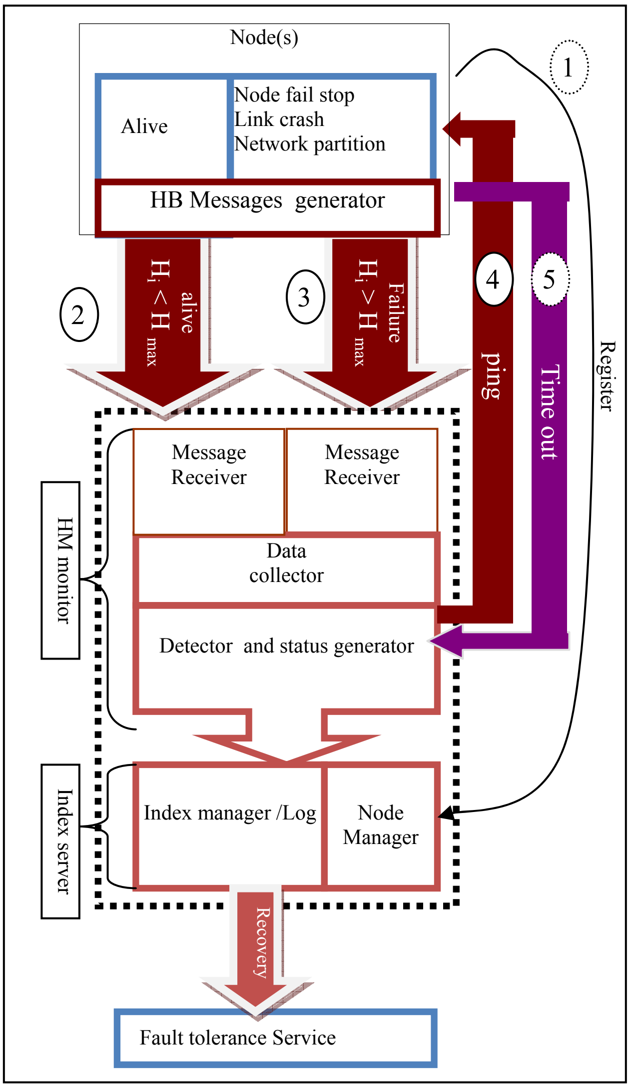

Figure1. Dynamic failure detection framework

1. A message receiver module in HBM receives heartbeat messages generated by message generator in each node. Each node has their own data receiver placed in individual storage in HBM server. This storage's identity is established by node ID. For instance, every time node *x* generates and sends a heartbeat message to HBM a folder for Data receiver allocated for the node in HBM will receive the message.

2. A data collector (DC) component in HBM fetches data from data receiver. The DC keeps track of all registered heartbeat messages and records whenever a heartbeat arrives.

3. A detector component in HBM is responsible for observing the state and identifying failed node(s) based on missing heartbeats. Since the detector has knowledge beforehand of the frequency at which heartbeats are being generated by a registered node, it can decide that the node is having missing heartbeats based on the contradiction with the previous heartbeats. It can interpret the messages embedded in previous heartbeats to resolve the state of the node and summarize the status information.

A. Dynamic Failure Detection

This section discusses failure detection mechanism and how DHF assigns the parameters and estimates the timing of heartbeat arrival. A member node and HBM server are connected by a communication channel. A Node n_i sends heartbeat messages to HBM. HBM analyses the messages to sort n_i's status.

In the following, the algorithm for node failure detection based on dynamic fault detection framework is given

```
Detect  new monitorable node Nx
/*check if Nx register with index log*/
IF Nx  in NM index
            then  update index manager  N++;
/*  HBw > Hmax allocated , may indicate node  problem*/
   if   HBw > HMAX
            ping node
            if ping timeout then
            node y failed;
            update index manager  N--;
            call fault tolerant service;
  else
             Hi = HMAX ;
  elseif
            Hi < HMAX
            then Hi = HMAX ;
  endif
```

Figure 2. Dynamic fault detection algorithm

HBM keeps and manages a list of time intervals of each heartbeat arrival B ={1.003s, 1.90s, 0.062s, 0.893s, 1.922s, 2.007s, …}. HBM refers to this list as one of the parameters to compute maximum waiting time HB_{max}. HB_{max} is a maximum inter-arrival time allocated for each heartbeat message extrapolated from the list B. Since network bandwidth and CPU load are unpredictable, this list is the key for HBM to dynamically tailor HB_{max} to adapt to changing conditions of the distributed system. HBM constantly reevaluates HB_{max} failure detection time to stay relevant to the current condition. This adaptive characteristic essentially reduces the false alarm rate. So we consider the maximum failure detection time to be the largest value in B:

$$HB_{max} = \{x | x \in B \text{ and } x = \text{largest value in } B\} \tag{1}$$

Along with list of B, HBM also stores the heartbeat interval Ti (in seconds) and the checkpoint Tc, the timestamp when the last heartbeat was received. This information is the core for HB detection to draw conclusions about a node status.

Basically for the most part, the sending times are influenced by the following environmental conditions;

Network delay Heartbeats time of arrival across a network is affected by network bandwidth and load. It is taken into consideration, that a heartbeat may be delayed due to congested network, therefore HB_{max} is adjusted accordingly.

HB message processing delay: a node taking longer to generate and send over a heartbeat, for example, due to processing overload or CPU busy. In many systems, the variations of the inter-arrival times due to processing delays of a node are negligible where this might not affect the HB_{max}. But when the variations are significant enough, HB_{max} would change accordingly.

Based on the data received, and the detection on comparison with HB_{max}, HBM will generate a suspicion status which indicates whether a node has failed or not.

B. Dynamic Interaction

A member node periodically sends messages which are its heartbeats within maximum time interval allowed. If the HBM does not receive the message after the maximum time interval elapses, the node is suspected to be having fault. Besides the prospect that CPU or network problem causing a node to be unable to send the heartbeat message, it could also be the node messaging/message generator having a problem. In this case the node is active and operational even though the node's heartbeat processing is having a problem. In this case, a pinging is necessary to invoke the reprocessing of a heartbeat. Integrating ping interaction into DHD can conclude the node status more correctly. Thus this technique is called dynamic heartbeat framework (DHF).

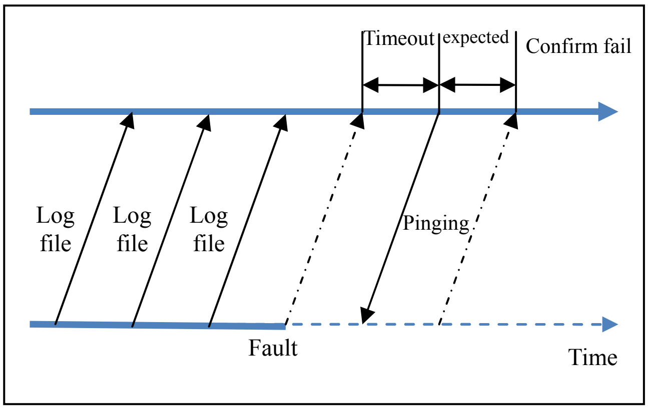

Figure 2. Ping with Heartbeat Mechanism

Figure 2 shows the flow of the fault detection service methodology with heartbeat mechanism. IF a heartbeat message exceeded the HB_{max} , the HBM will ping the member node (the node suspected to be having a problem). If the node does not reply and exceeds ping timeout, the node is considered to be in a failed state, the status of the node will be updated in the Index Server, and consequently, proper recovery action will be carried out.

III. PERFORMANCE

Currently, a widely used heartbeat monitoring technique is Globus heartbeat monitoring[23](GHM). In this technique, GHM receives heartbeat messages from member nodes at regular intervals.

The equation of a normal heartbeat operation without having any failure is given by:

$$\sum T_n = T_{HM} + T_i \tag{2}$$

where,

i) The number of nodes in a grid environment, n.
ii) The interval between each message sent to the heartbeat monitor, T_i.

iii) Time taken for a message to arrive at heartbeat monitor, T_{HM}.

The T_{HM} and T_i parameters are set to constants while the number of nodes in the grid environment, n is manipulated to measure the level of effectiveness of the mechanism used in detecting fault at different number of nodes. The number of nodes defines the size of the cluster. The inter arrival times of the heartbeat messages as well as time taken to detect failure of a grid component will be measured.

The GHM failure detection and DHF failure detection processes are demonstrated in Figure 3 and Figure 4 timeline diagrams respectively. A member node in GHM periodically sends heartbeat messages to its HBM.

If the HBM does not receive the message within the time interval, the application is considered to be having a problem. The HBM will wait for another interval to elapse. If the HBM receives the message within the next interval, the Index Server will be updated with the new status of the monitored node i.e. the application is OK. However, if the HBM does not receive the message after the second time interval, the application is declared as failed.

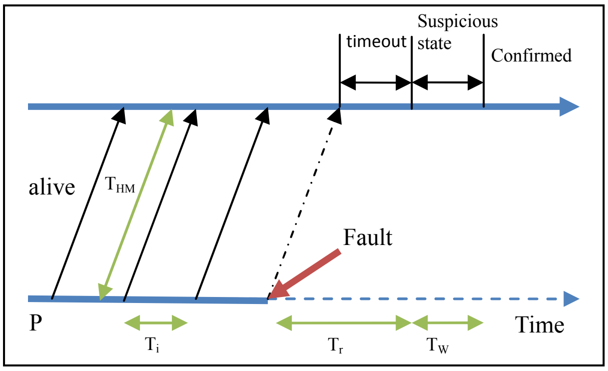

Figure 3. GHM failure detection

This will take longer time to detect failures and to prove that the node is faulty. The equation of FDS to detect a failure is given by:

$$\sum T_n = T_{HM} + T_i + T_r + T_W \tag{3}$$

where,

i) The number of nodes in a grid environment, n.
ii) Time taken for a message to arrive at heartbeat monitor, T_{HM}.
iii) The interval between each message sent to the heartbeat monitor, T_i.
iv) The timeout when the heartbeat monitor realizes that it has not received the message from the node, T_r.
v) The waiting time of the heartbeat monitor before declaring that the node has died, T_W.
vi) The time taken by heartbeat monitor to declare a node has died, $\sum T_n$

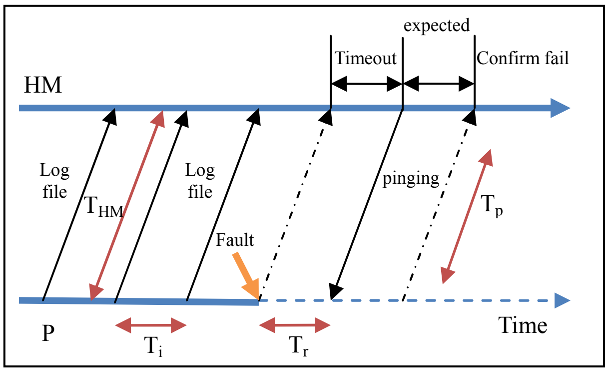

Figure 4. DHF failure detection

Figure 4 shows the DHF failure detection process that integrates the heartbeat mechanism with the ping mechanism in order to shorten the time taken to detect failure node(s) where waiting for the second interval to elapse is eliminated. The equation of the proposed model is given by:

$$\sum T_n = T_{HM} + T_i + T_r + T_P \tag{4}$$

where,

i) The number of nodes in a grid environment, n.
ii) Time taken for a message to arrive at heartbeat monitor, T_{HM}.
iii) The interval between each message sent to the heartbeat monitor, T_i.
iv) The timeout when the heartbeat monitor realizes that it has not received the message from the node, T_r.
v) The time taken by heartbeat monitor to ping out the suspicious node, T_P.
vi) The time taken by heartbeat monitor to declare a node has died, $\sum T_n$.

IV. Results Analysis

The failure detection performance result for GHM is illustrated in Table 1 and Table 2. Table 1 shows the time taken for declaring a failed node by GHM. Table 2 shows the overall time taken to detect the failed nodes within an environment where the number of nodes available varies from 10 to 100. Assuming that 10% of the numbers of nodes available are failed nodes.

TABLE 1.
THE TIME TAKEN TO DETECT A FAILED NODES BY GHM

	T_{HM}	T_i	T_W	T_r	$\sum T$
Time (ms-^{1})	30	100	100	20	250

TABLE 2.
THE TIME TAKEN TO DETECT FAILURES BASED ON NUMBER OF NODES, N IN THE ENVIRONMENT

N	100	200	500	600	700	1000
10% failure	10	20	50	60	70	100
$\sum T\ (ms^{-1})$	25000	50000	125000	150000	175000	250000

Table 3 illustrates the failure detection performance result by DHF while Table 4 shows the overall time taken to detect the failed nodes based on the environment where the number of nodes available varies from 10 to 100. With the assumption that 10% from the numbers of nodes available are failed.

TABLE 3.
THE TIME TAKEN FOR DETECTING THE FAILED NODES USING THE PROPOSED MECHANISM

	T_{HM}	T_i	T_r	T_P	$\sum T$
Time (ms^{-1})	30	100	20	24	174

TABLE 4.
THE VALUE OF N NODES AND TIME TAKEN TO DETECT FAULT USING THE DHF

N	100	200	500	600	700	1000
10% failure	10	20	50	60	70	100
∑ T(ms-1)	17400	34800	87000	104400	121800	174000

The GHM performance result obtaioned from Table 2 and DHF performance result from Table 4 are depicted in Figure 5 graph diagram.

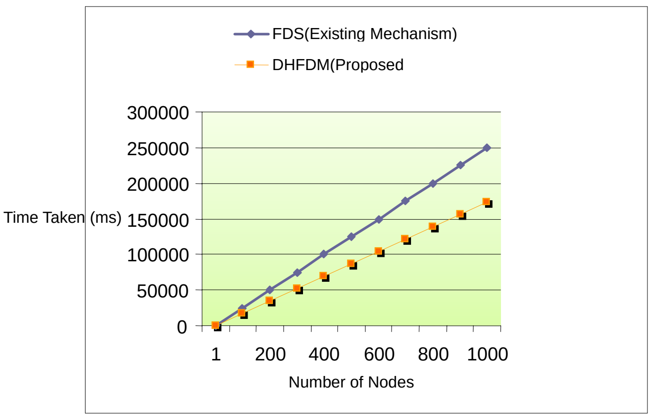

Figure 5. GHM and DHF performance comparison

Figure 5 show the performance result comparison between GHM and DHF. From this figure, when the number of nodes increases, the time taken to detect the failure nodes is increased. However, DHF required less time taken as compared to GHM because for each suspected failure node DHF does not need to wait for next interval. Alternatively, DHF pings the node to force the node to respond, whereas the GHM mechanism needs to wait for the second interval to confirm the failure state. For example from Table. 4, when the number of nodes is 70, DHF model needs only 12180ms of processing time while Table 2 shows that FDS model needs 17500ms. Thus, the result shows that DHF model provides an efficient approach up to approximately 30% better than GHM model.

V. Conclusion

This paper demonstrated a framework and algorithm in detecting failure within distributed environment for distributed system highly availability. The performance evaluation has been conducted between current GHM model and the proposed DHF model. The DHF is designed for monitoring current distributed. In this paper, DHF embeded pinging process into its framework and run at low level service thus it will never affect the system protocols and policy in order to avoid future problems. Embedding ping interaction into DHF is very significant in case occasion where a node cannot send the heartbeat message due to HB generator problem. Consequently, DHF can determine a member node status. The process also essential in reducing the detection time and false alarm rate. Futhermore, DHF always dynamically recalculates HB_{max} failure detection time based on a list of inter-arrival history data. Thus it can reduce the false alarm rate. Overall, This technique able to reduce the communication overhead by shortening the time taken to detect fault in a distributed environment.

References

[1] Fiaz Gul Khan, Kalim Qureshi and Babar Nazir. Performance evaluation of fault tolerance techniques in grid computing system Computers & Electrical Engineering 36(6) Elsevier B.V, (2010) 1110-1122

[2] Jesús Montes, Alberto Sánchez, María S. Pérez, "Improving Grid Fault Tolerance by Means of Global Behavior Modeling,", Ninth International Symposium on Parallel and Distributed Computing, (2010) 101-108

[3] Hwang, S. & Kesselman, C., Introduction, Requirement for Fault Tolerance in the Grid, Related Work. A Flexible Framework for Fault Tolerance in the Grid Journal of Grid Computing (2003) 251-272

[4] Christopher Dabrowski. Reliability in grid computing, Concurrency Computation: Practice and Experience. Wiley InterScience,(2009)

[5] K. Mills, S. Rose, S. Quirolgico M. Britton ,C. Tan An autonomic failure detection algorithm. SIGSOFT Softw. Eng. Notes, 29(1), (2004), 79–83.

[6] Matthew Gillen, Kurt Rohloff, Prakash Manghwani, and Richard Schantz.. "Scalable, Adaptive, Time-Bounded Node Failure Detection". 10th IEEE High Assurance Systems Engineering Symposium(HASE'07). IEEE Computer Society, Washington, DC, USA (2007)

[7] Stelling, P. Foster, I. Kesselman, C. Lee, C. Laszewski, G.: A Fault Detection Service for Wide Area Distributed Computations, In proceedings of HPDC, (1998) 268-278.
[8] Soonwook H.: A Generic Failure Detection Service for the Grid, Ph.D. thesis,University of Southern California, (2003).
[9] Renesse, R. Minsky, Y. and Hayden, M.: A Gossip-Style Failure Detection Service,Technical Report (1998), 98-1687
[10] Abawajy, J. H. and Dandamudi,S.P.: A Reconfigurable Multi-Layered Grid Scheduling Infrastructure, In proceedings of PDPTA'03), (2003) 138-144.
[11] Abawajy, J.H. Introduction. Fault Detection Service Architecture for Grid Computing Systems (2004)107-108.
[12] K. C. W. So, and E. G. Sirer, "Latency and Bandwidth-MinimizingFailure Detector", in Proceedings EuroSys(2007)
[13] L. Falai and A. Bondavalli, "Experimental Evaluation of the QoS of Failure Detectors on WAN", in Proceedings DSN 2005.
[14] A. S. Mohd Noor and M.M Deris. Extended Heartbeat Mechanism for Fault Detection Service Methodology. In D. Slezak et al. (Eds.): GDC 2009, Communication in Computer and Information Science, 63, 88–95, (2009)

Ahmad Shukri Mohd Noor is senior lecture of Computer science Department at Faculty of Science and Technology Universiti Malaysia Terengganu (UMT). Currently, he is a postdoctoral researcher with Distributed, Reliable and Intelligent control and cognitive Systems group (DRIS) at Department of Computer Science, Faculty of Science and Engineering, University of Hull. United Kingdom. He received B.Sc. in Computer Science from Coventry University 1997, M.Sc in High Performance System from University College Science and Technology Malaysia 1989 and Ph.D from University Tun Hussien Onn Malaysia 2012. His research interests include distributed computing, data grid and Application system development

Mustafa Mat Deris is a professor of computer science inthe Faculty of Information Technology and Multimedia, UTHM, Malaysia. He received a B.Sc. from University Putra Malaysia 1982, M.Sc. from University of Bradford 1989, England and Ph.D from University Putra Malaysia 2001. His research interests include distributed databases, data grid, database performance issues and data mining. He has appointed as one of the editorial board members for International Journal of Information Technology, World Enformatika Society, International Journal of Applied Science, Engineering and Technology (IJSET), International Journal of Electrical, Computer, and Systems Engineering (IJECSE), International Journal of Applied Mathematics and Computer Sciences (IJAMCS) and Encyclopedia on Mobile Computing and Commerce, Idea Group, USA; Apart from that he is also a reviewer for several International Journals such as Journal of Parallel and Distributed Databases,

Md Yazid Mohd Saman is Professor Computer Science Department at Faculty of Science and Technology Universiti Malaysia Terengganu (UMT). He received B.Sc from Essex university , M.Sc from UTM and Ph.D form Loughborough University, United Kingdom, his areas of expertise Distributed & Parallel Computing; Computer Network; Simulation & Performance Modeling.