DOI: http://dx.doi.org/10.21123/bsj.2017.14.4.0823

Network Self-Fault Management Based on Multi-Intelligent Agents and Windows Management Instrumentation (WMI)

Hasanen S. Abdullah [1] *Maha Abdulkareem Alrawi* [2] *Dalal N. Hammod* [3*]

Received 25/7/2017
Accepted 13/11/2017

Abstract:

This paper proposed a new method for network self-fault management (NSFM) based on two technologies: intelligent agent to automate fault management tasks, and Windows Management Instrumentations (WMI) to identify the fault faster when resources are independent (different type of devices). The proposed network self-fault management reduced the load of network traffic by reducing the request and response between the server and client, which achieves less downtime for each node in state of fault occurring in the client. The performance of the proposed system is measured by three measures: efficiency, availability, and reliability. A high efficiency average is obtained depending on the faults occurred in the system which reaches to 92.19%, availability 92.375%, and reliability 100%. The proposed system managed five devices. The NSFM implemented using Java and C# languages.

Keywords: Network Self-Fault Management, Intelligent Agent, Fault Detection, Fault Identification, Fault Recovery.

Introduction:

A network management system (NMS) is a set of tools of hardware and/or software that allows an Information Technology (IT) professional to supervise the individual components of a network within a larger network management framework [1]. The Autonomic management systems promises to provide guaranteed, smooth, and autonomous services of network and operations. There are four basics of self-X functions for autonomic computing [2]:

- Self-configuring – systems dynamically changing environments.
- Self-healing – systems diagnose, discover, and react to disruptions.
- Self-optimizing – systems tune resources and monitor automatically.
- Self-protecting – systems anticipate, detect, identify, and protect themselves from attacks from anywhere.

Network management is divided into five functional areas by the International Organization for Standardization (IOS) network management forum. The five functional areas of network management are:

Fault, Configuration, Accounting, Performance and Security that sometimes are described as FCAPS [3].

The fault and failure can be defined depending on the IEEE Standard Glossary of Software Engineering Terminology. A fault is a defect in a component or hardware device; but failure is the inability of a system or component to perform its required functions within specified performance requirements. A fault is the cause of failure and failure is the result of fault [4]. There are many achievements occurred in the field network self-fault management, each suggests a new method for developed network self-fault management. The most useful ones are mentioned in the following:

C. Schneider[5], (2015), proposed a new method for self-healing by using an unsupervised approach that combines performance tests to perform fault identification in an automated fashion with artificial neural networks, i.e. the correct and accurate determination of which computer features are associated with a given performance test failure. This approach uses three techniques of machine learning: Baum-Welch, Contrastive Divergence Learning, and Naïve Bayes. The Contrastive Divergence Learning minimized the human-

[1, 2] Department of computer science, University of Technology, Baghdad, Iraq.
[3]Department of computer science, University of Al-Nahrain, Baghdad, Iraq.
[*]Corresponding author:dal_scin81@yahoo.com

interaction beyond previous implementations by producing a list in decreasing order of likelihood of potential root causes which brings the state of the art one step closer toward fully self-healing systems. M. Toy [6] (2014), introduced the self-managed network that performs by identifying network failures and repair them, also the self-configurations network that is performed by configuration network resources and services. The architectures of the Self-managed Network Element (sNE) and Network Management System (sNMS) for centrally managed networks are described in this work. A hierarchy among repairing entities is defined. An in-band message format for Metro Ethernet networks is proposed for the fault management communication. In self-managed, when a single point of hardware is a failure, a network, isolating and identifying faults performed by itself and fixing them, and having technicians at the failure site only. Therefore, the operational cost reduced. D. Mitrovic, et; al.[7] (2010), present fault-tolerance to existing agent frameworks that an easy and a flexible way. The approach is using new two types of mobile agents; First, connection, Agent and Second Remnant Agent. The mobile agents manage efficient construction and maintenance of fault-tolerant multi-agent system networks, and implement a robust agent tracking technique.

Network Fault Management:

The detecting, diagnosing, repairing and reporting network equipment are the purpose of the fault management and efficiency of running network to keep from services failure. The functions of fault management are alarm surveillance, localizing the fault, management, testing, correcting the fault and trouble administration [8].

Fault management involves several steps: Data collection and modeling, Detection, Isolation, and Recovery, as explained in the following [9].

- Data collection and modeling: Errors can be reported by monitoring devices.
- Detection: Analysis the errors and define the type of errors.
- Isolation: Among the procedures/tools that aid in isolating a fault when an operational device suddenly fails.
- Recovery: Recovery actions are within the scope of external signaling for automated or manual correction of the problem.

Intelligent Agent:

Intelligent agent can be defined as software that acts on their behalf and assists people by allowing them to delegate work that they could have done. Agents can perform repetitive tasks, intelligently summarize complex data, remember things you forgot, learn from you and even make recommendations to you [10]. There are many characteristics of intelligent agent [11]:

- Autonomous, means that all actions of the agent have control of the agent.
- Goal-driven, means that an agent has a purpose, and represent in accordance with that purpose.
- Social, means that they interact, or communicate with other agents.
- Reactive, means that an agent senses for the dynamic environment and responds in a timely fashion to these changes.
- Customized or adaptive, means that an agent learns, or changes their behavior based on previous experience.
- Mobile, means that an agent move from machine to machine.

Windows Management Instrumentation (WMI) For Network Management:

Microsoft Windows operating systems run on local and remote computers for management data and functionality by the WMI. WMI management, data obtained directly through enterprise management tools such as Microsoft Operations Manager (MOM) and Microsoft Systems Management Server (SMS), or through scripts and applications. Scripts written in any scripting language can be used that can work with Windows Script Host. The WS-Management protocol can obtain WMI data through Windows Remote Management. WMI is the Microsoft implementation of Web-Based Enterprise Management (WEBM) , an industry initiative to establish standards for sharing and accessing management information over an enterprise network. WBEM provides the ability for the industry to deliver a well-integrated set of standard-based management tools, facilitating the exchange of data across otherwise disparate technologies and platforms [12]. WMI includes a CIM-compliant object repository and the CIM Object Manager. The object repository contains object definitions that supply data for managing hardware and software. Examples of WMI classes are the Win32 classes, such as Win32_Printer or Win32_ComputerSystem, and StdRegProv, which supplies registry data. The CIM Object Manager handles the collection and manipulation of objects in the repository and gathers information from WMI providers. WMI providers act as intermediaries between WMI and components of the operating system, drivers, applications, and other systems. [13].

The Performance Criteria:

The performance evaluation based on three evaluation performance metrics; these are efficiency, availability, and reliability [14].

1-Efficiency Criteria: Generally, The efficiency can be defined as the ratio between the output such as the amount of time, number of processes and the input such as the total amount of time, total number of processes. When efficiency improves, the output to input ratio improves. In order to measure the efficiency of the proposed system, the Up (the client work correctly) and the Down (the client work faulted) measures will be calculated by the equation (1):

Efficiency=output/input *100% … (1)

Where output represents the Up and Input represents all states (Up and Down).

2- Availability Criteria: Availability is the ratio of the uptime and the sum of the uptime and downtime of the system. This measure calculates the availability of the proposed system for the other clients. To measure the availability of the proposed system, the Uptime (amount of time when the system work correctly) and Downtime (amount of time when the system work faulted) measures will be calculated by the equation (2):

Availability=Uptime / (Uptime +Downtime)…(2)

3-Reliability Criteria: Reliability is the probability of the system to work accurately as a function of time 't', or is the ability of an item to perform a required function under stated conditions for a stated time period. In order to measure the service reliability of the proposed system, the Total Requests (total number of requests of the proposed system) and Successful Responses (the number of requests responded by the proposed system) measures will be calculated by the equation (3):

Service Reliability = (Successful Responses / Total Requests) * 100 …(3)

Proposed Method For Network Self-Fault Management:

Our proposed method supported devices related to two network services which are chatting system and video chatting. These devices are camera, monitor, soundcard, network card, and keyboard. When the fault management consists of three stages then the proposed method consists of three agents: the first agent used for performing the monitoring, intelligent to specific devices and trying to detect the fault in each monitored device, the second agent used for performing identify the type of fault based on the WMI. WMI is a part of windows that support 49 types of fault for each device, this agent returns a type of the fault; and the third agent used for performing the solution founding based on two phases: the first phase, search occurred fault in a database called the common faults that contain all the faults occurred previously. The second phase, used another database called recommendation solved for having a solution for the occurred fault. Fig. (1), illustrated the structure of the proposed network self-fault management.

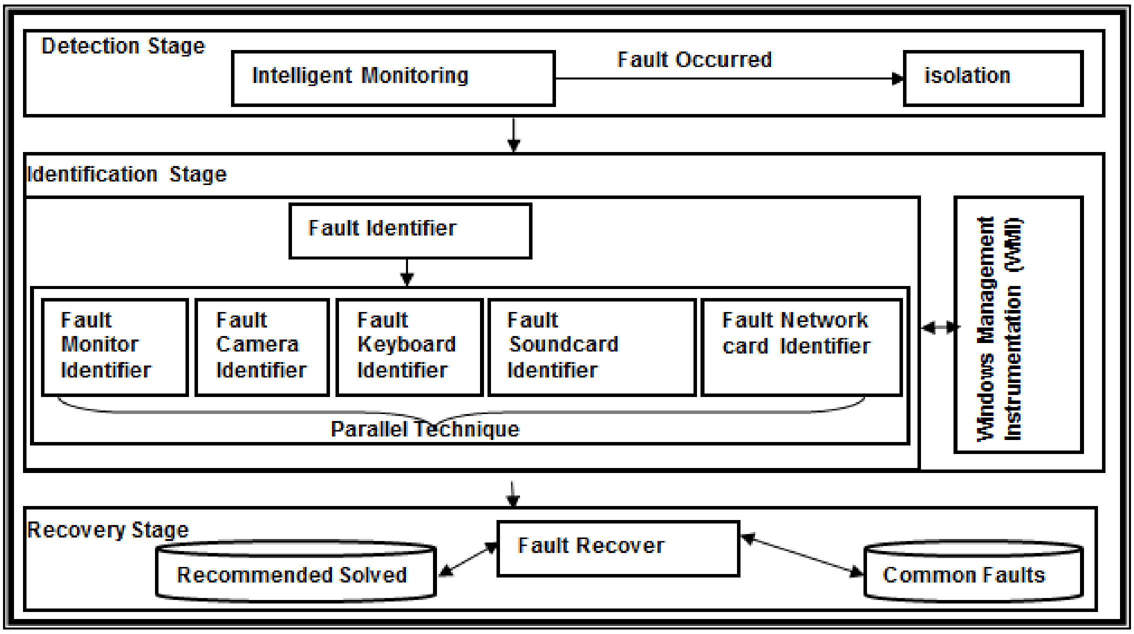

Figure 1. The Structure of the Proposed Network Self-Fault Management.

The following algorithm (1) explains the detection stage, where the monitoring process is continued until the fault occurred to perform the isolation process.

```
Algorithm (1): Detection Stage of the Proposed NSFM System
Input: Start Video  Chatting (Running Video Chatting System)
Output: Fault Occurred
Process:
Begin
Step1: Start Auto_Monitor Agent.
Step2: Not_Fault_Occured=True
Step3: While (Not_Fault_Occurred)
      Begin
          Step 3.1: Time=getSystem Time()
          Step 3.2: While (getSystemTime< >(Time+T)   //T represented an interval has 5 Sec.
 Begin
      Step 3.2.1: Get the Devices Status :
      Step a: Camera_Status=Get Camera Status()
      Step b: Network_Card_Status=Get Network Card Status()
      Step c:  Soundcard_Status=Get SoundCard Status( )
      Step d:  Monitor_Status=Get Monitor Status()
      Step e: Keyboard_Status=Get Keyboard Status( )
   Step3.2.2: IF ((Camera_Status=False) || (Network_Card_Status=False) || (Soundcard_Status=False) || (Monitor_Status=False) || (Keyboard_Status=False))  Then
           Not_Fault_Occured=False
   End While
  End While
    Step4: Isolation Node
    Step4.1: Close the Connection.
    Step4.2: Remove the Client from Clients List.
End.
```

Algorithms of Identification Stage:

Identification stage consists of five secondary agents, each agent is responsible for identifying the fault of a specific device by using WMI. The steps of the created WMI class and obtained information explained in the algorithm (2). Identification stage is explained by the following steps:-

```
Algorithm (2): Identification Stage of the Proposed NSFM System
Input: Fault Occurred.    /*The output of algorithm (1) */
Output: Device's Name, Device's Type  and Type of Fault.
Process:
Begin
Step1:Fault Identifier Agent Created and Started.
Step2:Fault Identifier Agent Created  Five Sub-Agent.
Step3:Each Agent Responsible for a Specific Managed Device and Running in Parallel with other agents.
Begin
Step3.1:Parallel.Invoke
Begin
Step a: Obtained Information from WMI Class (PNPEntity)
Step b: Obtained Information from WMI Class(NetworkAdapter)
Step c: Obtained Information from WMI Class(SoundDevice)
Step d: Obtained Information from WMI Class(Desktop_Monitor)
Step e: Obtained Information from WMI Class(Keyboard)
  End.
End.
    Step4: Return Fault Information for each managed device.
End
```

Algorithm of Recovery Stage:

Algorithm (3) explains the recovery stage of the proposed NSFM system, where the Device Name, Device Type and Type of Fault are three input variables obtained from identification stage and the Fault Solved represents the solution to the fault occurred. Algorithm (4) explains the proposed Network self-fault management approach.

```
Algorithm (3): Recovery Stage of the Proposed NSFM System
Input:  Device Name, Device Type and Type of Fault. /* The output of the algorihm (2)*/
Output: Fault solved.
Process:
Begin
Step1:Fault_Found=False
Step2:While (Not ( Fault_Found))
    begin
   Step 2.1: Flag=False.
   Step 2.2: Flag=Search (Device Name, Device Type, TypeOfFault) in Common Faults. // This is the first phase
   Step2.3:IF (flag=True) Then
               Fault_Found=True
        Step2.4: Flag=Search (Device Name, Device Type, TypeOfFault) in Recommended Solutions. // This is the second phase
        Step2.5:  Fault_Found=True
     End While
      Step3: Return "The Solve of  each Fault".
End
```

```
Algorithm (4): The Proposed NSFM Approach
Input: Starting Video Chatting System (Running Video Chatting System)
Output: Fault Solved
Process:
Begin
Step1: Call::Detection Stage of the Proposed NSFM System;
         //algorithm (1)
Step2:Call::Identification Stage of the Proposed NSFM System;
        //algorithm (2)
Step3: Call::Recovery Stage of the Proposed NSFM System;
       //algorithm (3)
End
```

Discussion and Test Results:

This section displays an evaluation of the proposed NSFM that obtained faster method than other methods because client tried to solve his/her faults alone without server action, therefore, the proposed method avoids the bottleneck problem that all clients ask the server and wait for receiving responded and reduced load of the network traffic. The NSFM consists of three intelligent agents that running automatically and has a full control of it. The first agent, is used for the detection stage of the NSFM system. In this stage the agent worked continuously monitoring for occurring faults and made isolation for the node (the computer of the Client) from NSFM system when the fault occurred. The second agent, is used for the identification stage of the NSFM system. In this stage the agent is responsible for five sub agents that worked in parallel for identifying the fault occurred. The third agent, is used for the recovery stage of the NSFM system. In this stage the agent is responsible for finding a solution depends on two phases to solve the fault occurred.

Evaluation of the Efficiency Criteria

The proposed system applied on WLAN which consists of a set of nodes that denoted as Nn= {N1, N2, N3….. Nn}, each node has one of these status at one time: Up state (Working correctly) denoted as 1 or Down state (Working Failure) denoted as 0. Four video conferences used for testing the NSFM system. In each video conference testing the node state (UP state or Down state) for 5 seconds (for each period of time) and recording the values ("1" or "0"). When finishing the video conference, calculated the total number of upstate, total uptime, total downtime, and recording the request information such as total number of requests and successful request. Table (1) shows the efficiency of the results through video conference 1, video conference 2, video conference 3, and video conference 4, where N1 denoted Node1, N2 denoted Node2, and No. of up denoted number of up state. The efficiency criteria for each node in the video conference is calculated by using the equation (1). Also, the efficiency calculated at each node at a specific moment of time.

Table 1. The Efficiency of the Proposed System for Each Node in VC1, VC2, VC3, and VC4.

	VC1				*VC2*				*V C 3*				*VC 4*			
Time (Sec)	*Node 1 (N1)*	*Node2 (N2)*	*No. of Up*	*Efficiency at each time*	*Node 1 (N1)*	*Node2 (N2)*	*No. of Up*	*Efficiency at each time*	*Node 1 (N1)*	*Node2 (N2)*	*No. of Up*	*Efficiency at each time*	*Node 1 (N1)*	*Node2 (N2)*	*No. of Up*	*Efficiency at each time*
5	1	1	2	100%	1	1	2	100%	1	1	2	100%	1	1	2	100%
10	1	0	1	50%	0	1	1	50%	1	0	1	50%	1	1	2	100%
15	1	1	2	100%	1	1	2	100%	1	1	2	100%	1	1	2	100%
20	1	1	2	100%	1	1	2	100%	1	1	2	100%	1	1	2	100%
25	1	1	2	100%	0	1	1	50%	1	1	2	100%	1	1	2	100%
30	1	1	2	100%	1	1	2	100%	0	1	1	50%	1	1	2	100%
35	1	1	2	100%	1	1	2	100%	1	1	2	100%	1	1	2	100%
40	1	1	2	100%	1	1	2	100%	1	1	2	100%	1	1	2	100%
Total N0. Of the Up (output)	8	7	-	-	6	8	-	-	7	7	-	-	8	8		
Efficiency for each Node	100%	87.5%	-	-	75%	100%	-	-	87.5%	87.5%	-	-	100%	100%		
The average of Efficiency	93.75%				87.5%				87.5%				100%			

From the results in Table (1), high efficiency is obtained when the number of the up state is increased. Noticeable decrease in the efficiency occurs when the up state is decreased and the down state is increased. The VC1 shows the smallest value of No. of Upstate (7) and have a low efficiency (%87.5), and the largest value of No. of Upstate (8) achieves high efficiency (%100). The average efficiency for the VC1 is equal to the (%93.75). The average efficiency of VC2 is approximately equal to (%87.5) because the efficiency of the node 1 is equal to (%75) and the efficiency of the node 2 is equal to (%100). The smallest value of No. of upstate achieves of the node1; therefore, node1 has a low efficiency and a large value of No. of upstate achieves of the node2; therefore, node2 has a high efficiency. The average efficiency of VC3 is equal to (%87.5) because the efficiency of the node 1 is equal to (%87.5) and the efficiency of the node 2 is equal to (%87.5). The two nodes have the same values of the No. of upstate; therefore, the two nodes have the same efficiency too. Also, the video conference 4, the optimal state for the proposed system when the two nodes: node 1 (N1) and node 2 (N2) have the upstate for long time of video conference then each of the nodes has a high efficiency (%100) and the average of efficiency in the video conference 4 is equal (%100). Fig. (2) plotting the average efficiency for the four video conferences of the proposed system.

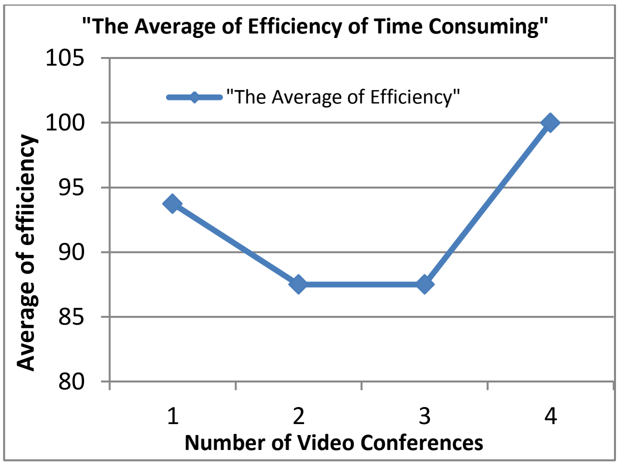

Figure 2. The Average of Efficiency of the NSFM System

Evaluation of the Availability Criteria

When using the Availability measure for the proposed system by using equation (2), the uptime and downtime are calculated for each node in the video conference. Table (2) shows the results of availability of the proposed system, where the total time represents the period of time, No. Node represents the number of nodes, Uptime represents the time for corrected working, and Downtime represents the time for solving the fault of the node, where the availability depends on the amount of uptime and downtime for each node in a specific video conference.

Table 2. Availability for each node in a specific video conference.

VC No.	*Total Time (Sec)*	*No. Node*	*Uptime (Sec)*	*Downtime (Sec)*	*Availability*	*Average of the Availability*
VC1	40	N1	40	0	1	0.94
		N2	35	5	0.88	
VC2	40	N1	30	10	0.75	0.875
		N2	40	0	1	
VC3	40	N1	35	5	0.88	0.88
		N2	35	5	0.88	
VC4	40	N1	40	0	1	1
		N2	40	0	1	
		The average of the Availability				92.375

From the above result in Table (2), In the first row, the node 1 has availability equal to (1) because the node 1 has the uptime (40 Sec) and the downtime (0 Sec). But the node 2 has the availability equal to (0.88) because the node 2 has the uptime (35 Sec) and the downtime (5 Sec) of the total time of the video conference is (40 Sec). Better results could be seen of the availability of the two nodes when the first node (N1) and the second node (N2) have value equal to (1), while the results show a noticeable decrease in the availability when the downtime is increased. Also, the large value of downtime (10 Sec) and have a low availability (0.75), and the smaller value of downtime (0 Sec) and have the high availability (1). The average availability of the system is approximately equal to (%92.375). Fig.(3) explains the availability of NSFM system.

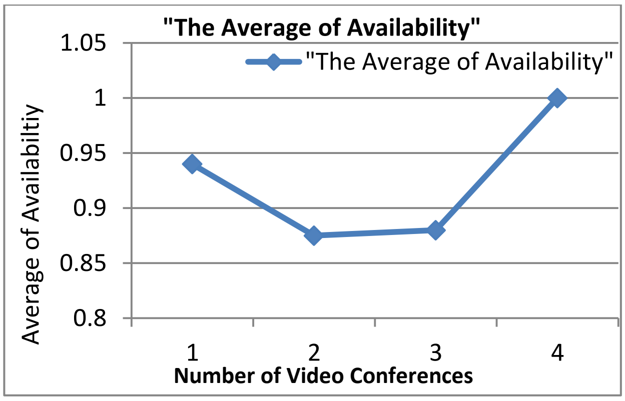

Figure 3. Availability of the NSFM System

From the above fig. (3), where the x-axis represents the number of video conference and y-axis represents the average of the availability, in the first video conference, the average of the availability is (0.94), the second video conference, the average of the availability is (0.875), the third video conference, the average of the availability is (0.88), and the forth video conference, the average of the availability is (1).

Evaluation of the Reliability Criteria

When using a Reliability measure for the proposed NSFM system, the total request for the node and number of successful responses are calculated. Table 3 shows the Reliability of the proposed NSFM System for the node, where the total request denoted to the total number of requested to solve the occurred fault, and the successful responses represents number of solved requested successfully.

Table 3. Reliability of the Proposed network self-fault management System

VC No.	*Successful Responses*	*Total Requests*	*Reliability*	*Average of Reliability*
VC1	2	2	100	100
	2	2	100	
VC2	1	1	100	100
	2	2	100	
VC3	1	1	100	100
	1	1	100	
VC4	1	1	100	100
	1	1	100	
The average of the Reliability				100

From the result in Table (3) displayed high reliability of the proposed system for the nodes (100) because the client responded to each request by himself/herself without any server action. The average reliability for each video conference is equal to (100). Fig. (4) explains the Reliability of the proposed NSFM system for the node.

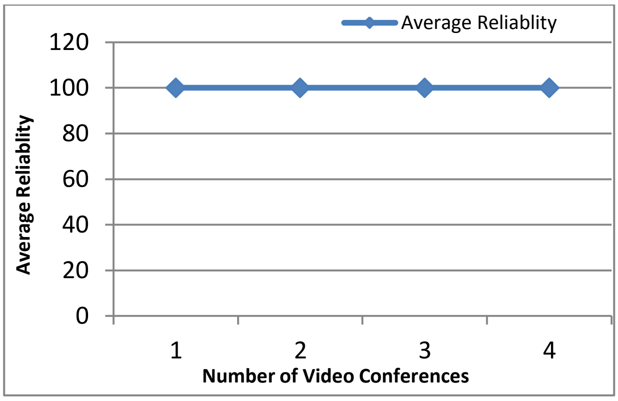

Figure 4. Reliability of the Proposed NSFM System

The traditional system required a collection of main faults for building a database or it is using the distribution management with a middle level of the administrator. The test results of the proposed work, show a fast and efficient to be applied into the network services because it is using intelligent agents and parallel technique for reducing detection time, identification time and recovery time. Also, the proposed work using the WMI and it does not need a built-in a special database.

Conclusions:

The proposed NSFM system used three intelligent agents with full self-auto control; the first agent is used for the detection stage, the second agent is used for the identification stage, and the third agent is used for the recovery stage. The intelligent agents give the system more flexiblity and powerful properties to ensure finding the solution to the fault when it occurs. self-management reduces the managing traffic in the network (no bandwidth-intensive client/server message exchange). Most operations of the proposed NSFM system management are done without server intervention. The property enables the proposed ssystem to reduce the congestion in the network which supports offering the services without noticeable delay. Video chatting is represented as a video conference application. The proposed NSFM system successes in accomplishing this application with high performance criteria in term of efficiency, availability and reliability. The proposed NSFM system optimizes the efficiency criterion which reaches 92.19%, availability criterion 92.375%, and reliability criterion 100%.

References:

[1] Adebayo, B. S.; and Kolo, M. I. 2012. Agent-Based Faults Monitoring in Automatic Teller Machines. Computing, Information Systems & Development Informatics Vol. 3 No. 4 , September.

[2] Mearns, H. 2012. CARMA: Complete Autonomous Responsible Management Agent (System). PhD. dissertation in University of Technology, Sydney.

[3] Kotsopoulos, K. 2010. Managing Next Generation Networks (NGNs) Based on the Service-Oriented Architecture (SOA). PhD. dissertation in University of Bradford.

[4] Ryschkewitsch, M. G. 2012. Fault Management Handbook. National Aeronautics and Space Administration (NASA).

[5] Schneider, C. 2015. Using Unsupervised Machine Learning for Fault Identification in Virtual Machines. PhD. dissertation in University of St. Andrews.

[6] Toy, M. 2014. Self-Managed Networks with Fault Management Hierarchy. Elsevier.

[7] Mitrovic, D.; Budimac, Z.; Mirjana; and Vidakovic, M. 2010.Improving Fault-Tolerance of Distributed Multi-Agent System with Mobile Network-Management Agents. Proceedings of the International Multiconference on Computer Science and International Technology, P.P. 217-222.

[8] Gavalas, D.; Greenwood, D.; Ghanbari, M.; and O'Mahony, M. 2010. A Progressive Network Management Architecture Enabled by Java Technology. Communication Networks Research Group.

[9] Dolisy, J. 2015. 5 elements of advanced network monitoring. available at: https://gcn.com/articles/2015/12/07/advanced-network-monitoring.aspx

[10] Briola, D. 2012. Negotiation in Multiagent Systems: Protocols, Ontologies and Applications. PhD. dissertation in Universit`a di Genova, Italy.

[11] Shoham, Y.; and Leyton-Brown, K. 2010. Multiagent Systems Algorithmic, Game-Theoretic, and Logical Foundations. Shoham and Leyton-Brown.

[12] Wilson, E. 2006. Microsoft Windows Scripting with WMI Self-paced Learning Guide. Microsoft Press.

[13] Russinovich, M.; Solomon, D.A.; and Ionescu, A. 2012. Windows Internals Part 1. Microsoft Press, 6th edition.

[14]Paul Barringer, H. 1997. Availability, Reliability, Maintainability, and Capability. Beaumont, Texas.

الادارة الذاتية للأخطاء في الشبكة بالاعتماد على نظام تعدد الوكلاء الذكي وادارة نوافذ الادوات القياسية

حسنين سمير عبدالله [1]   مها عبدالكريم الراوي[2]   دلال نعيم حمود [3]

[1]، [2] قسم علوم الحاسوب، الجامعة التكنولوجية، بغداد، العراق.
[3] قسم علوم الحاسوب، كلية العلوم، جامعة النهرين، بغداد، العراق.

الخلاصة:

هذا البحث يهدف الى اقتراح طريقة جديدة لنظام الإدارة الذاتية للاخطاء في الشبكة (NSFM) تعتمد على تقنيتين هما تقنية الوكيل الذكي لغرض دعم تلقائية مهام ادارة الخطأ والتقنية الثانية هي تقنية ادارة نوافذ الادوات القياسية (WMI) التي تحدد الخطا بشكل اسرع وبغض النظر عن نوع الموارد المستخدمة (الاجهزة). يعمل النظام المقترح لادارة الخطا على تقليل عبء سير وتبادل البيانات عبر الشبكة من خلال تقليل الطلب والاستجابة بين الخادم والزبون مما حقق وقت توقف اقل في حالة ظهور خطأ عند الزبون.

تم قياس اداء النظام المقترح من خلال ثلاثة مقايس : الكفاءة و التواجدية و الموثوقية. تم تحقيق معدل كفاءة عالية بالاعتماد على الاخطاء الحاصلة في النظام وصل الى (92.19%)، و التواجدية (92.375%) و الموثوقية (100%). وقد تمكن النظام من ادارة خمسة انواع من الاجهزة وقد تم استخدام لغة الجافا و#C.

الكلمات المفتاحية: ادارة الاخطاء الذاتية في الشبكة، الوكيل الذكي، اكتشاف الخطأ، تحديد الخطأ، ايجاد حل للخطأ.