Health Evaluation of B/S Information Management Software System

Xiaona Xie[1, a *], Zhengwei Chang[2, b] and Wei Jiang[3, c]

[1]Control Engineering College, Chengdu University of Information Technology, Chengdu, Sichuan, China

[2]State Grid Sichuan Electric Power Research Institute, Chengdu, Sichuan, China

[3]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China

[a]xnxieok@163.com, [b]changzw@126.com, [c]weijiang@uestc.edu.cn

Keywords: B/S architecture; Health evaluation; Information management software; Maintenance.

Abstract. B/S information management software systems have been widely used in power grid corporations. For the purpose of maintaining system efficiently and accurately, condition based maintenance should be applied. In this paper, a health evaluation system of B/S information management software is designed. A health metric based system state partition approach is proposed. Based on the results of stressing test using Loadrunner on a real information system, the quantified performance metrics and the corresponding evaluation model are established and applied into an information management system of power Grid Corporation. Finally, experiments validate the efficiency of this health evaluation system.

Introduction

With the popularization of computers, especially the development and application of database technology, the information management system changes enormously. Information management system alters the daily work, and improves the efficiency of the enterprise greatly. Currently the B/S based system follows the dynamic web technology, adds the development concept of information management system software. The system not only fully meets the requirements of network management, but also becomes the technology of first choice for modern management information system [1]. Its main advantages are better openness, less difficult to manage, simpler maintenance and easier operability.

B/S architecture, which adopts the browser and the server architecture, is a kind of improved C/S structure [2]. B/S architecture focuses the core part of system functions on the server, while the client only needs to install a browser, thus reducing the cost of system maintenance and upgrades. B/S architecture has become the mainstream of enterprise information management systems [3].

State Grid Sichuan Electric Power Corporation has implemented SG186 information system projects of comprehensive application. The scale of the information system is expanding rapidly. With the new system and new information technologies are applied, the possibility of system failure is also increased, and the system stability and system manager's requirements are also improved. The new approach, i.e. condition based maintenance has to be focused on information system. How to assess the health of information management software effectively and ensure the stability of the system software has become the most concern problem. In this paper, we design a software system health evaluation approach for B/S based information management system.

Health Indicator Constitution of B/S Information Management System

The health status of B/S based information management system consists of the client`s health indicators, the Web server health indicators and the databases and database host server health indicators.

Client Health Indicator Form. System client is the browser accessing content, and its health indicators consist of the response time, the error rate and the security.

Response time, which is the time from customers click the log interface of browser to the information management system respond and return to the normal client interface; Error rate can be measured by the count of client interface occurs when user accesses browser; Security can be measured by the count of that the user can`t log in when inputting the correct commands.

Web Server Health Indicator Form. The most important part in B/S architecture is the application server and Web server [4]. This part is the most prone to failure place. Web and application servers situate in the middle layer, the intermediate layer can be logically divided into three layers: presentation layer, business logic layer and data interface layer. The corresponding data evaluation indicators of three layers mainly are: server queue, server threads, server memory and the number of JMS sending messages. From the server`s physical structure and user experience, the data evaluation indicators which we can refer to mainly are: CPU utilization, disk utilization, memory utilization, server response time.

Server queue exists in each component, if the queue is overloaded, it will result in deadlock and cause the server to enter a dangerous state. Server threads indicate the operation number of execution queue can simultaneously perform, it represents the system`s concurrent degree. When the number of threads is more than a certain value, memory exhaustion phenomenon will occur, while the time spent on the thread switch will increase. Only in the case of server memory is sufficient, the server can provide services. If we continues to increase consumption memory, it will cause the server cannot process the request timely and be likely to cause a buffer overflow. JMS (Java message service), is the API for message-oriented middleware (MOM) on Java platform [5], used for sending messages between two applications or in a distributed system`s processes asynchronous communication. In normal circumstances, the message sending operation can be carried out. If you do not increase the JMS server in a warning status [6], which may cause the server cannot process the request message in time and may likely cause message overflow.

Database and Database Hosting Server Health Indicator Form. Data layer is mainly responsible for storing corporate data, user data and etc. It also provides an interface for the application server. In the B/S architecture, the data layer is mainly composed of two parts: the database itself and the database`s host server. These two parts complement each other and determine the performance and health status of the external data layer together.

Host server health status consists of three modules to reflect: the resources owned by the server, the server port opening status and the performance of the external server performance (with the user experience related).

For the host server health measure, we use a modular, hierarchical thought. The advantage of this approach is clear and intuitive, but also consistent with the idea of software design and software testing. We place factors that influence each other inside a module, so that the association between the module and the module is little by which they can have a high cohesion and a low coupling.

There are 4 main performance parameters which resources interest in: CPU usage; memory usage, virtual memory usage; disk space usage, I / O speed; database connection usage. Performance parameters that performance status interests are mainly three: database throughput, response time of the database, the competition resources that the database wait events before 10 involve in.

Oracle database performance is stable, powerful, and able to meet the requirements of the vast majority of enterprise-level [7], so this paper uses Oracle Database as example, other databases can refer to similar evaluation. Oracle Database mainly consists of Database and Instance [8-9], this paper based on the two constituent elements divides Oracle Database health into database instance health status module and database storage health status module.

There are 4 main parameters referred by the health state of database instance, they are: the hit rate of data buffer area, the hit rate of log buffer area, the hit rate of share area and the hit rate of sort area. There are 4 main parameters referred by the health state of database storage, they are: the using rate of file system, the using state of list, the using state of exchange area and the using state of database dump area.

State Partition Based on Testing

Testing Analysis Based on Loadrunner. This paper adopts the instance testing based on Loadrunner [10]. Its test process contains 5 basic procedures. The controller invokes pressure test at first. Then it gives testing orders to the virtual user creator. After that the creator starts simulating a large amount of real users` data, where the pressure comes. When the whole system receives the pressure, it starts to generate kinds of performance states. State surveillance will capture these states in real time, and feed back to the controller immediately. These testing results will be collected and stored by monitor so that they can finally be used to produce the analysis report [11].

Here we provide an example which is the SG186 safety surveillance and manage business system built by State Grid Sichuan Electric Corporation. They build a testing environment for B/S information management system. The system consists of several modules; each module supports few operations such as: read files, publish files, delete files, modify files, give review opinions, statistics reading times and sort by date, etc.

The first job for testing is the selection of testing cases which should obey the standard below: typical business process, users operate frequently, larger effect on system performance, performance testing pressures compliance the real operates of business system users. According to these standards, we select main modules and design each operation as a job. Finally we write the testing cases.

After analysis of user login operate module, we have testing results below: the average login time value is 5.4 seconds, which is in a proper range. However some of values among them have a large difference compared with the average value. Such as the response time had reached to 18.7 seconds when the time was 12:48 which means the system was in danger state. At this time the using rate of CPU is 86%. According to large amount of testing experiments` data, we analysis the relations between the user experience and the actual using rate of CPU. Finally, we partition the state of CPU into 3 levels: normal, warning and danger, shown in Table 1.

Table 1 CPU states

Index	Normal	Warning	Danger
Using rate of CPU (%)	<30	30 ~ 80	>80

State Partition. In this paper, we will use test software called Loadrunner. It simulates large amount of user operates and gains lots of testing data of states. By analyzing the relations between the user experience and the health of the system, we finish the health state partition for servers and clients, shown in Table 2 and Table 3.

Table 2 State partition of client

Index	Normal	Warning	Danger
Response time(s)	0~15	15~30	>30
Error rate (%)	<10	10~60	60~100
Security(times)	0	1	2

Table 3 State partition of application server (web server)

Index	Normal	Warning	Danger
Server queue (%)	0~85	85~90	90~100
Server thread (%)	0~80	80~90	90~100
Server memory (%)	0~80	80~90	90~100
Message send by JMS	0 ~ 3	3 ~ 5	>= 5

Similarly, according to the testing results, we finish the health state partition to databases and their hosting servers, shown in Table 4 and Table 5.

Table 4 State partition of database

Index	Normal	Warning	Danger
Using rate of file system (%)	<80	80~95	>95
Using rate of list (%)	<85	85~97	>97
User_dump_dest (%)	<90	90~95	>95
Core_dump_dest (%)	<90	90~95	>95
Background_dum_dest (%)	<90	90~95	>95
Using rate of exchange area (%)	<80	80~95	>95
Using rate of data buffer area (%)	90~100	60~90	<60
Buffer NOwait ratio (%)	99~100	90~99	<90
Redo NOwait ratio (%)	99~100	90~99	<90
Latch Hit ratio (%)	99~100	90~99	<90
In-Memory Sort hit rate (%)	99~100	90~99	<90

Table 5 State partition of database hosting server

Index	Normal	Warning	Danger
Jobs per second	10~300	0~10	0~10
conversation waiting for resource	0~10	10~20	>20
Database waiting time ratio (%)	<10	10~20	>20
Database using time ratio (%)	<80	80~90	>90
CPU using rate (%)	0~80	80~95	>95
Memory using rate (%)	0~80	80~95	>95
Disk using rate (%)	0~80	80~95	>95
Disk active time ratio (%)	0~70	70~95	>95

Comprehensive Evaluation and Testing of Software System

Comprehensive Evaluation of Information Management Software. According to the health module and the testing results, we can completely measure the health state of whole B/S software system. In this paper, we adopt a method by grading.

First state: whenever one of the state`s data is in danger state, the whole system will be defined in danger state, and must be disposed immediately.

Second state: when all states in B/S system are not in danger state (in normal state or warning state). At this time the system need to be estimated properly.

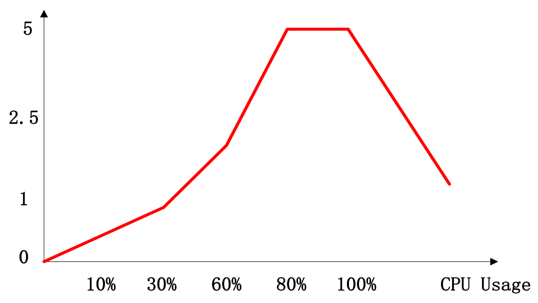

Figure 1. Data index grading rules

Here we will describe the dynamic grade deduction method of text by using an example of CPU. According to Table 1, when the using rate of CPU is in [0, 30%], we can estimate CPU as normal state.

For more rational grading, we adopt dynamic grade deduction mechanism. As the increasing of the using rate of CPU, the grade deduction will also increase dynamically. And in different partition the changing degrees of grade deduction are also not the same, shown in Fig. 1.

As we can see, the grade deduction for each data state is no more than -5. For example, when the total deduction is -30, that is to say there are at least 6 (or more) states are working in high load mode or full load mode. We define this state as attention state which should be optimized.

Summary

This paper establishes evaluation model based on health indicators by simulating the stress tests of real environment, the entire software system`s health status is determined by the client`s health status, Web server and application server`s health status, database and its hosting server`s health status.

We finish dividing the states of the software system and also establish grading evaluation model using the method of dynamic grading and grade accumulating deduction. Then we deploy it in SG186 systems. According to the experimental results shows that the software system is in a normal state, thereby we verify the effectiveness of the health assessment system.

Acknowledgements

This work is supported by a project grant from State Grid Sichuan Electric Power Corporation, Artificial Intelligence Key Laboratory of Sichuan Province (Grand No. 2015RYY04) and research startup foundation of Chengdu University of Information Technology (Grand No. KYTZ201520).

References

[1] D. Zhao, W.D. Zhao, The safety production information system based on B/S structure, China Public Security: Academic Edition, 4(2007) 97-99.

[2] Y. Liu, W. Zhang, Z. X. Wang, Embedded remote monitoring system based on B/S and C/S structure, Instrument Technique and Sensor, 11(2009)39-41.

[3] J. Peng, D.R. Chen, Implementation based on ebXML registry of B/S structure, Computer Applications, 25(2005) 236-237.

[4] D. Alur, D. Malks, J. Crupi, et al, Core J2EE Patterns(Core Design Series): Best Practices and Design Strategies, Sun Microsystems Inc, 2003.

[5] R. Johnsom, J2EE development frameworks, Computer, 38(2005)107-110.

[6] V. Matena, B. Steams, L. DeMichiel, Appling enterprise JavaBeans: component-based development for the J2EE platform, Pearson Education, 2003.

[7] Q. Hu, General maintenance analysis of Oracle database, Technology and Living, 13(2011)102.

[8] H.L. Zhang, General maintenance analysis of Oracle database, Computing CD-ROM software and application, 4(2011)67.

[9] G.X. Li, F. Wang, G.R. Li, Maintenance of Oracle database in hospital information system, Information of Medical Equipment, 19(2004) 32-34.

[10] P. Yang, J. Li, Using LoadRunner to test Web’s load automatically, Computer Technology and Development, 17(2007) 242-244.

[11] Y. Li, G.X. Zhou, A performance testing workflow solutions research and design based on LoadRunner, Application Research of Computers, 26(2009) 4143-4115.