Test Information Collection System Design and Implementation of a Distributed Environment

Nan Ding

Department of Mechanical and Electrical Engineering,
Hebei Vocational &Technical College of Building Materials,
Qinhuangdao, 066000, China

snma@163.com

Abstract

Software testing is an important part of the software development process, is used to confirm the quality or performance of a program is in line with the development of before have put forward some requirements. Is software testing in software before put into operation, the software requirements analysis, design specification and coding of the final review, is the key step in the software quality assurance. Software testing is to find errors and execution process. Software testing in software life cycle across two stages: usually when writing out after each module is for it to do the necessary tests (called unit tests). Coding and unit test belong to the same phase of the software life cycle. After the end of this stage to all kinds of comprehensive testing of software systems also, this is another independent stage in software life cycle, namely the test phase. Software test information collection under distributing environment is one of most important test manage research, the paper adopt DCOM groupware developing technology, make transfer to different platform and type, and use OOD mode to design a test collection software system.

Keywords: *Distributing environment; DCOM technology; Object Oriented*

1. Introduction

With the development of computer technology, the software scale more and more huge, the function is becoming more and more complex, software testing is an important means to ensure software quality and reliability, however, to test the Linux, WINDOWS system and other different heterogeneous database platform and ORACLE, Sybase, such comprehensive systems often need to use a variety of tools, test testing techniques and methods, saves manpower and time cost is large, the control and management of the testing process and evaluation of test results are very difficult. At present, there are many foreign software manufacturers to develop a variety of software testing management products, like Rationale's SQA Manager Products, Compuware Director products, these products are testing tool for each factory management, cannot achieve cross-platform operation, for other manufacturers products are not compatible with the management.

This article describes the test information collection system will use the communication facilities between the set of DCOM objects, object technology application module and object-based communication, to achieve unified the different platforms, different types of testing tools management and call.

2. Summary of Test Data Acquisition System

The system for collecting and managing the test information to test management environment as the center, through its provide test management environment and integrated access engine, realize the integration and test tools, test case management subsystem, database management subsystem. It mainly composed of three major components: the client components, the test host components, test management server components. The system architecture is shown in Figure 1.

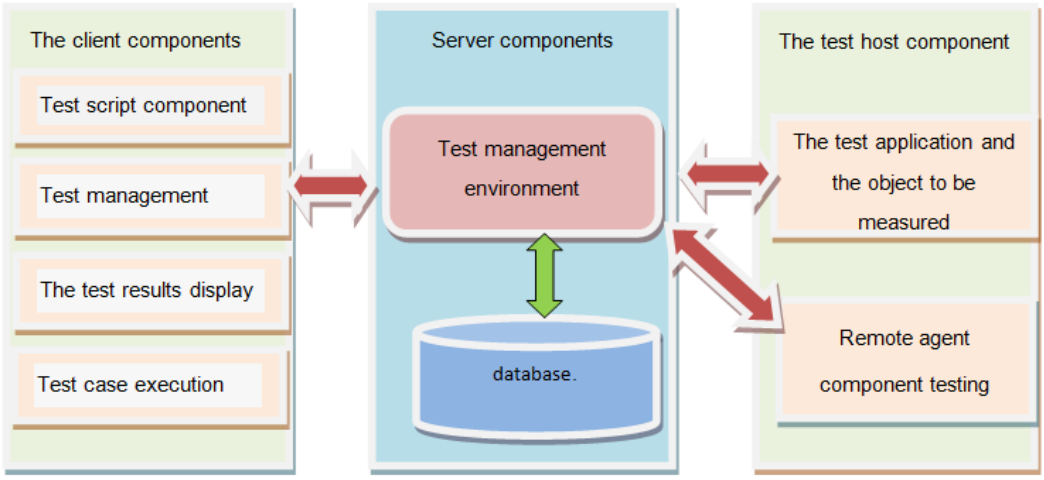

Figure 1. System Structure Diagram

（1）The server component is a Web server, when the network client to connect to the Web server through the browser existed in the ActiveX or DLL forms components are downloading data from the server to the client.

（2）The test host component by the test tool remote agent components, test application and the object to be measured. The remote agent is multithreaded components, tools running, can view the progress and status of the test case execution through the remote agent, data collect test results and update the data server through API data. Test tools for remote agent component and API component of the COM.

（3）The client components by the testing management console, test tools and test tools script management controls execution result controls, they are ActiveX controls. Use a browser to open a test page from the server automatically downloaded to the client browser.

3. The Design Scheme of Test Data Acquisition System

3.1. Software Process Design System

The structure of software test data acquisition system consists of test information acquisition system client program, the test client and host test. The Client Deployment browser, users use the browser connects to the server, the completion of various stages of testing process, query, modify the configuration.

Test management service server deployment test information collection system service layer, database access engine, the test information acquisition system client access through

the DCOM service program testing information acquisition system need to get the data, capability of the client can complete the function completely depend on the test information collection system service program, all data read and save operation done by him. Test information collection system service program does not access the database directly, but through the database access engine to achieve access to different types of database, database access engine and database are independent of each other, the two can not on the same computer. Testing process management database data quantity depends entirely on the test of the size of the project, the type of database to support Oracle, Access, Sybase, *etc*... Save the test process in the necessary test data, test data, test project management personnel, test configuration, test script and test results of the test process management database.

If the test case type tool testing, when executing the test cases, the client program testing information acquisition system of the test configuration and test agent sent to the testing tool to test the host agent, by the test tool agent started testing tools, perform the appropriate testing, test results generated the specified data, test results Converter Analysis test result data, generation of test management database to identify the format, test tools agent will test result parse the data stored in the test database management, test information acquisition system client display test status and results. This is the system software automatically processes. See Figure 2.

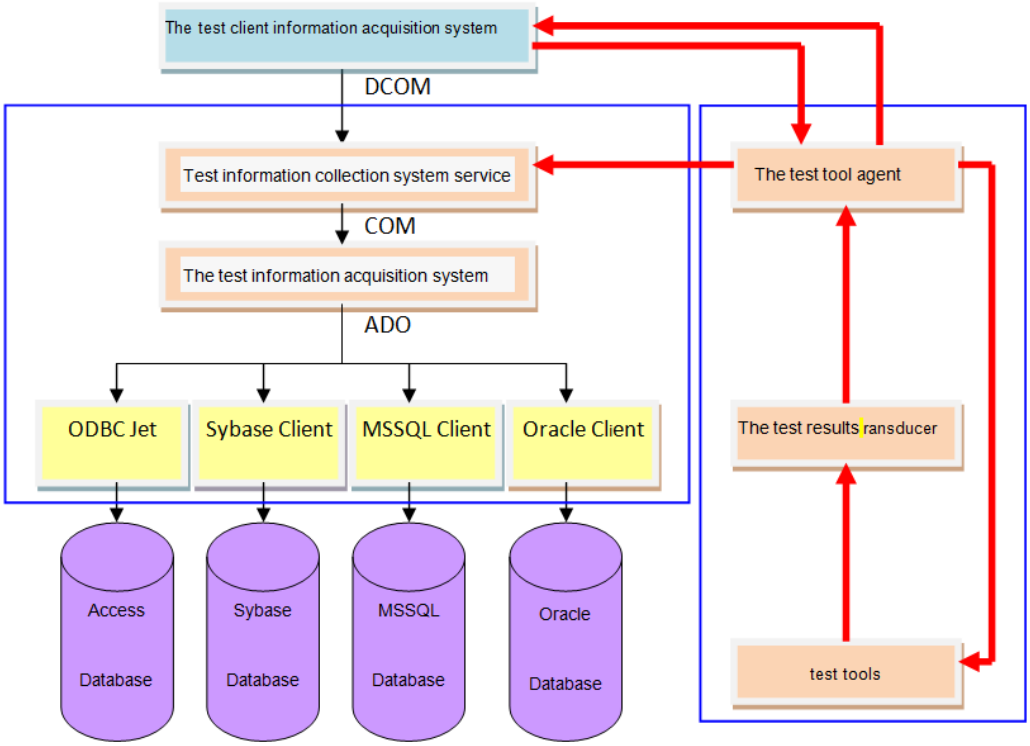

Figure 2. Flow Chart of Software Implementation

3.2. System Interface Design

A transducer assembly **i**ncludes a host of the test tool, the component is used to extract the test tool from the specific test tools generate test scripts, test configuration and execution

results associated with the test data, and converting it to identify the format. Table 1 describes the system interface and several major.

Table 1. The Main Interface

Interface name	Interface parameters	return value	description	Location for Connector
IORunTest	strTestID----out strRunID----out strCycleID----out strTestType----out	True False	Test data acquisition system to test the client host sends the implementation of test case information, return have no received signal	test agents
IOGetTestInfo	strTestID----in strScriptPath----out	0 1	The test agent through the interface, test case	Test management server
IOSaveTestResult	strTestID----in strResultPath----in strTestResult----out	0 1	Return whether the data success	Test management server
IOTestStatus	strTestStatus----out	True False	Test status query	The test agent

4. Realization of Test Data Acquisition System

Embedding COM component class management software, which include testing tool integrated entrance. The software uses the COM component class in the configuration file ID, ensure the system started to refer to the class, and the class references other assemblies, embedding management tool control component.

Figure 3 is a way to solve this problem. Agent Interface is an interface, provide operational testing.

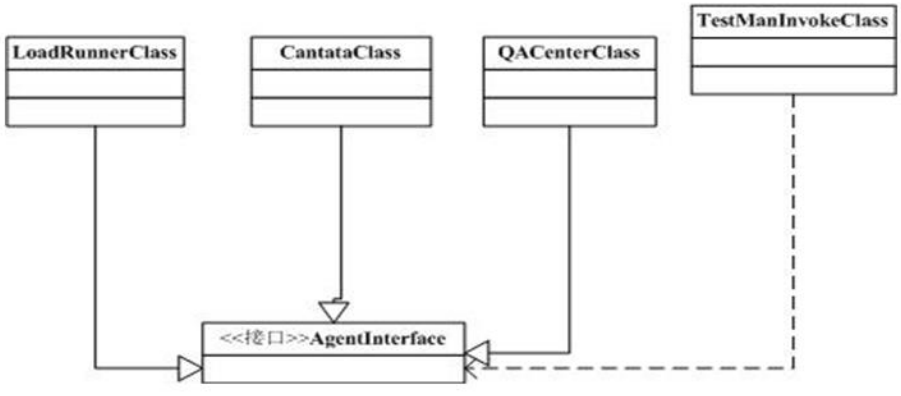

Figure 3. Schematic Diagram of Tool Integration Interface

Load runner, Cantata required methods; implementation is completed in succession in the Agent Interface class. Agent Interface can be the following code:

```
Public interface Agent Interface {

 Public abstract void Start ();

 Public abstract void Stop ();

 Public abstract void Pause ();
```

Interface provides a simple startup (Start), stop (Stop), and the suspension (Pause) test method. The LoadRunnerClass code is as follows:

```
 Public class LoadRunnerClass implements AgentInterface {

 Public void Start (){

     // the concrete realization of the code.}

     Public void Stop (){

     ---

 }
```

TestManInvokeClass's operations, responsible for instantiating an object, and the operation, the call object, code is as follows:

```
 Public class TestManInvokeClass {

 Private AgentInterface testToolAgent;

 Public void init (){

     TestToolAgent = new LoadRunnerClass ();

}
```

The init function of the realization of class Roadrunner. If the new added a test tool: NewTestTools. Can be achieved as follows:

```
  Public class NewTestToolsClass implements AgentInterface {

  Public void Start (){

     //The concrete realization of the code}

  Public void Stop (){

     ---

}
```

The TestManInvokeClass class code:

```
Public class TestManInvokeClass {
Private AgentInterface testToolAgent;
Public void init (){
    TestToolAgent = new NewTestToolsClass ();
}
}
```

From the above analysis we can see that, if the new test tools, only need to increase the successor to the Agent Interface of the Class. Frame structure does not need to change the code, changing the structure is very clear. In order to test tools Cantata as an example, the main structure based on test information process platform of Unix Socket 4 as shown in Figure 4.

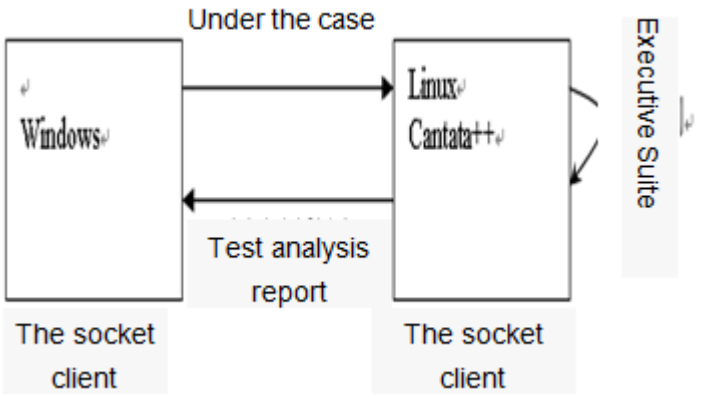

Figure 4. Cantata Integrated Structure Diagram

Cantata is a support C/C++ language, object-oriented, testing tool also supports a plurality of operating system, the main functions are: dynamic testing, static analysis and coverage analysis, support integration testing and regression testing. It provides a command line and integrated IDE environment.

The Socket server uses the Linux Socket programming, the Socket client uses the Windows Socket programming. Send test cases in Windows end, let the test cases to run correctly in the Linux end of the Cantata, after the execution of test cases, using Cantata report generation tool to generate reports in XML format, finally obtained via Windows and view the test analysis report.

5. Concluding Remarks

With the continuous development of network technology, a large number of distributed systems are deployed in the network, common, such as Web applications, Web services, *etc.* Different from general distributed system software system, it usually consists of multiple

subsystems of physical distribution, these subsystems through mutual cooperation computing task, characteristics of physical distribution, concurrent access, is sensitive to timing, platform heterogeneous, etc. In addition, the many subsystems in a distributed system in such aspects as trigger mode, working way is often does not belong to the same type, for example, protocol messages through the network subsystem and subsystem of using local interfaces to method calls, *etc*. Testing of distributed systems, therefore, also different from general software system test, need to adopt the distributed testing.

For distributed test, test process is a kind of high activity to process control request, therefore system need timely access to global state in order to correctly guide process; Secondly, in the process of testing, the system must be able to easily monitor and control the testing process. Therefore, distributed test system is suitable for adopting the centralized distributed strategy, that is, several controlled computer controlled by a central computer, the whole test process, and resource management are done by center, it distributed to master the whole test the state of the environment, so as to send control commands.

Distributed test environment with strong activities in all process, one step of operation failure can lead to the entire testing process interrupt and abnormal, thus need a stable communication environment. At the same time, communication is mainly between the center node and the execution node and two nodes of the main work is focused on testing activities and in the center of the logical node and the execution node parallel each other, have certain independence. Therefore, distributed test system for providing services of distributed system, suitable for use based on message communication way.

Distributed software testing has become a kind of the most important means of computer software testing work, we adopt the DCOM component technology, to realize the automatic acquisition distributed environment of different software testing management and testing information, the software has been applied to the testing work, the next step will be to extend its function, to achieve a variety of test tools integration.

Acknowledgements

The work in this paper has been supported by Hebei province education department, Fund number is (JYGH2011016).

References

[1] J. D. McGregor and D. A. SYKES, “Object-oriented software testing”, Beijing: CITIC Publishing House, **(2002)**.
[2] M. Fester and D. Graham, “Software test automation technology and detailed examples”, Beijing: Publishing House of electronics industry, **(2000).**
[3] J. Newkirk and R. C. Martin, “Extreme Programming in practice Chinese version”, People's Posts and Telecommunications Publishing, **(2002)** June.
[4] M. Haran, A. Karr, A. Orson, *et al.*, “Applying classification techniques to remotely-collected program execution data”, Lisbon, Portugal: Proceedings of the International Symposium on the Foundations of Software Engineering, **(2005)**.
[5] B. Mediated and Y. Atif, “Context-based matching for Web service composition”, Distributed and Parallel Databases, vol. 21, no. 1, **(2007)**, pp. 5-37.
[6] Liyongke, “A universal automatic test system based on Configuration Technology”, Computer measurement and control, no. 8, **(2005)**.
[7] T. Abdellatif and F. Boyer, “A node allocation system for deploying JavaEE systems on Grids”, Hammemet, Tunisia, (**2009)**.
[8] A. K. Bharti and S. K. Dwivedi, “E-Governance in Public Transportation: U.P.S.R.T.C. – A Case Study”, Kathmandu, Nepal, (**2011)**, pp. 7-12.
[9] S. Z. C. S. ChangChun, *et al.*, “A Novel Two-stage Algorithm of Fuzzy C-Means Clustering”, **(2010)**.
[10] CHINA, G.C.O.M., “The trust model based on consumer recommendation in B-C e-commerce”, **(2011)**.

[11] Y. Jiansen, *et al.*, "Suspension K&C Characteristics and the Effect on Vehicle Steering", **(2010)**.
[12] C. Juan, *et al.*, "Semi-physical simulation of an optoelectronic tracking servo system based on C MEX S functions", **(2010)**.
[13] X. Cao, W. Kou, X. Zeng and L. Dang, "Identity-based anonymous remote authentication for value-added services in mobile networks", IEEE Trans. on Vehicular Technology, vol. 58, no. 7, **(2009)**.
[14] H. Yang, F. Ye, Y. Yuan, S. Lu and W. Arbaugh, "Toward resilient security in wireless sensor networks", Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing, **(2005)** May 25-28; Urbana-Champaign, IL, USA.
[15] T. M. Connolly and C. E. Begg, "Database systems: a practical approach to design, implementation, and management", Addison-Wesley, **(2009).**

Authors

Nan Ding received her Master's degree in Computer Science and Technology from Yanshan University of Technology in Hebei, China Her research interest is mainly in the area of Data mining and software testing she has published several research papers in scholarly journals and international conferences in the above research areas.