TESTING WEB APPLICATIONS INTELLIGENTLY BASED ON AGENT

Lei Xu and Baowen Xu
Department of Computer Science & Engineering, Southeast University, Nanjing 210096, China
Jiangsu Institute of Software Quality, Nanjing 210096, China

Keywords: Web Application Testing, Capture-Replay, Intelligent Agents

Abstract: Web application testing is concerned with numerous and complicated testing objects, methods and processes. In order to improve the testing efficiency, the automatic and intelligent level of the testing execution should be enhanced. So combined with the specialties of the Web applications, the necessity and feasibility of the automatic and intelligent execution of the Web application testing are analyzed firstly; Then, on the base of the related work, the executing process of the Web application testing is detailedly described and thoroughly analysed, so as to determine the steps and flows of the testing execution along with the adopted techniques and tools; next, improve the capture-replay technique and make it fit for the dynamic characters of Web applications, and adopt the intelligent Agent to realize the monitor, management and exception-handler of the whole testing execution. Thus, in this way, the process of the Web application testing can be implemented automatically and intelligently.

1 INTRODUCTION

Web applications can realize the information sharing in the wide world, so multiple users pay attention to the new comings and the user number increased at a rare bat. Thus the testing for these Web applications must be carried out systemically, completely and sufficiently, so as to ensure their qualities and satisfy users' requirements finally. However, since Web applications have the characters such as distributed, dynamic, multi-platform, interactive and hypermedia, and their running environments are heterogeneous and autonomous, it is more difficult to carry out the testing tasks. So in order to enhance the testing efficiency and improve the testing effect, appropriate testing methods and tools are needed, which are focused on these specialties.

Web application testing is concerned with the multiple page contents, testing targets, steps and methods, and the whole process generally requires a great deal of manpower and material resources. For the sake of reducing the testing cost, the automatic level of the testing process should be improved, i.e. go along with the testing under the tools' assistants, make use of tools to dispose the routine problems of testing and the testers only monitor the whole process. At the same time, the related tools must have a certain degree of intelligence so as to adapt to the dynamic and evolutive specialties of Web applications.

At present, researchers pay more and more attentions to Web testing, propose some methods and obtain many important results (Gao, J. et al., 1999),(Kallepalli, C., and Tian, J., 2001),(Kung, D.C. et al., 2000),(Liu, C.H., 2002),(Ricca, F., and Tonella, P., 2000),(Warren, P., Boldyreff, C., and Munro, M., 1999). And we also attempted some research in the area of Web testing modelling, performance improvement, and the detailed testing methods (Xu, L., and Xu, B.W., 2003),(Xu, L. et al., 2003),(Xu, L. et al., 2003). This paper focuses on how to realize the Web application testing automatically and intelligently. So in Section 2, we introduced the related work so as to explain the necessity and feasibility of carrying out the Web application testing automatically and intelligently. In Section 3, we detailedly described and thoroughly analyzed the executing process of the Web application testing, so as to determine the steps and flows of the testing execution along with the related contents such as the adopted techniques and tools; Furthermore, we presented the realization techniques of the testing process, i.e. improved the capture-replay technique and made it suitable for the dynamic characters of Web applications, and

Xu L. and Xu B. (2005).
TESTING WEB APPLICATIONS INTELLIGENTLY BASED ON AGENT.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 194-199
DOI: 10.5220/0002540501940199

adopted the intelligent Agent to realize the monitor, management and exception-handler of the whole testing execution. Section 5 is concerned with the conclusions and the future work.

2 RELATED WORK

Testing Web application is concerned with the contents of the three aspects, i.e. the running environment, previous preparing work and the executing process. Since Web applications are used by vast users, the testing is usually carried out in the client-side so as to have more authenticities, and the testing is also demanded to be executed under all kinds of typical equipments of client-side. In the preparing work before testing, testing objects and testing plans must be determined, which include generating and selecting the test cases (including the input and expected output of the testing), and establishing the testing actions and steps. So in the executing process of the testing, testers often carry out many test cases under several typical testing environment so as to obtain the coverage rate as high as possible, and they execute the actions due to the previous determined testing steps, such as the testing input, function running and results comparison with the expected outputs.

Manually executing the Web application testing has high cost and low efficiency. So automatic testing is needed obviously. However, the exceptions are often produced in the testing process, and these errors could exist in the server side, network transfers or the browser parser. Furthermore, the presented error messages are very vague, and this brings great difficulties to the error diagnosis and exclusion. Thus it is urgent to improve the intelligent degree of the testing so as to guarantee the execution of the testing.

On the other hand, since the testers generally carry out the testing with the rigid steps and the determinate contents, the whole process is suitable for the automatic execution. By now, there have been several testing assistant tools, which are applied in the performance testing or the regression testing. Paper (Anupam, V et al, 2000) described the working process and related techniques of the capture-replay tools, i.e. firstly capturing the actions carried out by users in the correlative pages, then storing them to the appropriate places due to a certain sequences and contents, and replaying the steps and contents after the searching and matching finally. Apparently, this technique can be used to realize the automatic testing. However, there are many dynamic-generated pages in the Web applications and the changes are very frequent in the pages, so the intelligent degree of the capture-replay tool also should be enhanced.

The intelligent Agent has its background in the early work on artificial intelligence (AI) when researchers concentrated on creating artificial entities that mimicked human abilities (Hewitt, C., 1977), and has good attributes such as autonomy, cooperation and learning (Krulwich, B., 1997),(Murugesan, S., 1998). So they can operate on themselves without the manual guidance and can perform certain tasks on behalf of its users. It could also interact with other intelligent agents and/or human in performing its task(s) via some communication language. A key attribute of an intelligent Agent is its ability to learn to understand the user's preferences and behaviour, to cope with new situations it may face and to improve its performance over time. These characters make the Agent rather fit for the distributed platform such as the Internet, and the Agent has made effective progress (Belkin, N.J. and Croft, W.B., 1992),(Guttman, R.H. et al., 1998),(Wooldridge, J., and Jennings, N.R., 1995) in Information Retrieval (IR), personalization services, and etc. Furthermore, the technique of the Agent also can be applied to the intelligent Web application testing (Xu, L. et al., 2003).

Thus we can combine the capture-replay technique and the intelligent Agent to the automatic and intelligent testing for Web applications, i.e. manage the capture-replay tools by the Agent, improve the testing effect by the learning function of the Agent so as to deal with the real changes, exceptions and faults, and realize the intelligent search and fuzzy match to fit for the dynamic and interactive properties of Web applications.

3 TESTING SCHEME FOR WEB APPLICATION

Executing the Web application testing should consider the related factors completely, and this process is divided into three parts: testing input, testing execution and testing output. As shown in Figure 1, the input part of the testing is the test cases that are prepared previously, which includes the input information, expected output and the detailed

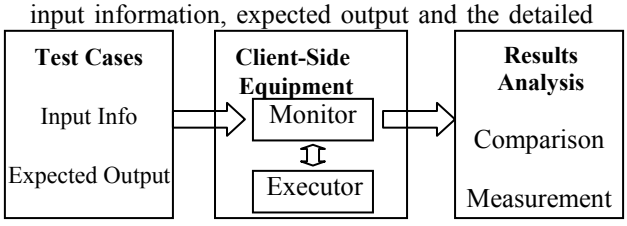

Figure 1: Execution Process of the Web Application Testing

test steps; then under a certain client-side equipment, the testing monitor organizes and manages the executors to fulfil the testing tasks; next, output the testing results to the last phase, and analyze and compare the results with the expected output, furthermore give metrics and feedbacks to the Web application.

The precondition of the Web application testing is the test cases generation and selection, so it is important to ensure the quality and quantity of the test cases. The usual steps of generating and selecting test cases can be described as follows:

(1) In order to obtain the executing sequences of the testing, the testing requirements and the structure contents of the Web application are considered and analyzed in detail, so as to determine the testing steps and input information such as variable, parameter name, type and value domain.

(2) For ascertaining the expected output, the general method is analysing the Web application and the testing requirements, and running the test cases several times in advance when the dynamic scenes are considered, then classifying the results into different kinds so as to provide references to the next testing.

(3) Based on the generated test cases, further work such as the simplification is needed to obtain the right number of test cases that have the equal qualities.

The quality of the test cases influences the testing effects directly, thus testers usually spend more testing costs during this process of test case generation. In paper (Xu, L. et al., 2003), we proposed a test case generating method based on the combinatorial testing and apply this method to the Web compatibility testing. In this way, we can cover more situations with less number of test cases and obtain higher testing efficiency.

The executing part of testing is constituted of client-side equipment, monitor and executor. Among them, the client-side equipment provides the testing environment, which has a certain degree of typicality and is more important in the Web compatibility testing; the monitor is the kernel part of the testing execution, and it obtains the test cases directly, controls and organizes the whole testing tasks, monitors the running status of every executor and guarantees the load equilibrium and communication among the executors, and can deal with certain exceptions, which is carried out by testers or some intelligent entities; the executor is the main actor who fulfils the testing tasks, and it executes the test cases due to certain sequences and steps heavily and tediously, i.e. based on the testing steps and input information in the test cases, the executor visits the Websites or pages under test, fulfils the prescribed testing actions, then outputs the testing results to the next phase. Furthermore, the expected outputs in the test cases are also delivered to the next phase by the monitor, so as to process the results comparison and feedback.

The usual testing process and steps are described in the above, and the executing efficiency is the main problem to solve. As shown in the before, the testing for Web applications has huge scale and tedious tasks, and it is impossible to execute the testing only by manual work. So the testing must be fulfilled under the assistant tools to realize the automatic or semi-automatic testing; and at the same time, the intelligent level of the related tools also should be improved so as to deal with the exceptions.

After obtaining the testing results, the comparison with the expected results is demanded so as to judge whether pass the testing. Quantitative analysis is still needed to the testing results. Then based on the existed criterions and standards, the Web application metrics can be established and adjusted so as to evaluate the quality of the Web application and the efficiency of the testing. Furthermore, the modification and maintenance to the Web application are required due to the testing results, i.e. feedback the testing results to the Web application itself, and further improve the system's quality, performance, reliability, usability and safety.

4 REALIZATION TECHNIQUES

Based on the previous work, in order to realize the automatic and intelligent Web application testing, adjustment and optimisation are also needed to the capture-replay technique and the intelligent Agent, so as to be seasoned with the interactive, dynamic and frequent-change properties of Web applications, and finally complete the testing tasks successfully.

4.1 Obtaining Testing Behaviours

In general, the capture-replay technique is used to realize the automatic Web application testing. Therefore, we must capture the testing behaviours of testers firstly, then record these steps and actions by Script language, and finally replay the Script to fulfil the automatic testing. In our method, we set an Agent in the test side, and capture the contents such as executing actions and input information of the testers by the recording, learning and analysis functions of the Agent. Since Agent has the ability of self-learning, we can capture more precise testing behaviours by the Agent's coordinate and analysis.

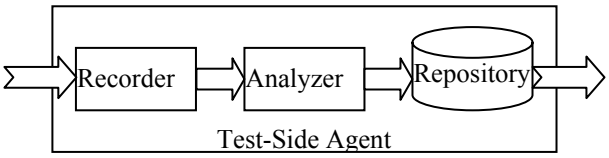

Figure 2: Inner Structure of Test-Side Agent

Figure 2 shows the inner structure of the test-side Agent, which is consisted of three portions, i.e. the recorder (Krulwich, B., 1997), the analyzer and the repository. This Agent runs at the executing circumstance of the testing, and the flow steps can be described as follows:

(1) When the testers begin their testing actions, the recorder inside the Agent starts executing its tasks at the same time, and it captures and records all the behaviours that have taken place in sequence, such as the hyperlink-hitting and the form submission, so we can capture testers' executing sequences and all the testing actions in focus, and gain a full executing process of testing at last.

(2) After a period of time (set to the requirement), the analyzer in the Agent carries out the classification and coordinate work to the contents inside the recorder, i.e. classifying the testing actions due to the different testing targets (for example, in the form function testing, testers must prepare the test cases that have all kinds of selected-values; and in the Web performance testing, multiple virtue users are needed to visit the server concurrently).

(3) In this way, the testing behaviours of the testers are captured, which are focused on different testing targets. And such knowledge is stored in the repository for the coming usages.

Furthermore, the above process can be refreshed due to the real instances of the tester's executing actions, so as to insure the contents in the repository have the precision and the freshness.

Since the tasks of Web application testing are very heavy and tedious, it is impossible to capture all the testing scenes; additionally, the dynamic generated pages and the frequent changes make the captured testing scenes unable directly use or never be applicable. However, the captured testing behaviours obtained in our method have better operability owing to the test-side Agent, for by the learning and analysis of the Agent, we can present the corresponding testing process due to the different testing targets, thus they have wider adaptabilities. Furthermore, since the testing process is recorded very detailedly, we have the specific guidance abilities on the testing details.

4.2 Executing Testing Automatically

After capturing the testing behaviours through the intelligent Agent, the automatic replay for the test can also be carried out by Agent, i.e. choosing some intelligent Agents as the testing executor, and substituting the testers to fulfil the testing tasks, so as to improve the testing efficiency evidently. Thus the capture-replay task can be implemented by the test-side Agent and the test-execution Agent respectively, but the two kinds of Agents have some differences in the inner structure and the functional realization.

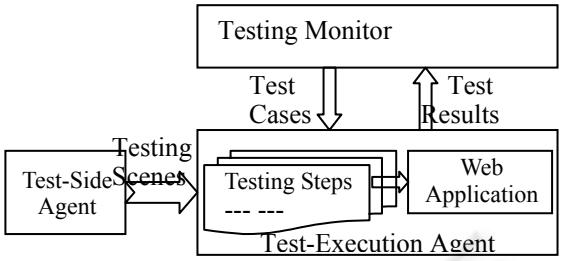

Figure 3: Workflow of Test-Execution Agent.

The workflow of the test-execution Agent is shown in Figure 3, and this kind of Agent performs the testing tasks under the control of the testing monitor. The working process of this Agent can be described as follows: firstly, based on the type of the testing target, choose the corresponding testing scene in the test-side Agent's repository; then determine the testing steps and executing behaviours, combined with the input data that are provided by the test cases; next, the test-execution Agent visits the Web pages under testing and implement the recorded actions such as hitting hyperlinks or filling in the form information; finally, output the testing results to the testing monitor, by which we can enter the next testing phase.

When the test-execution Agent replays the testing behaviours, some problems may take place such as unable using the recorded actions, since there are dynamic generated pages and often-changed pages in the Web applications. In order to solve these problems, fuzzy match and intelligent identification techniques are fetched so as to suit for the properties of Web application. Namely, as the different pages are concerned, which are generated dynamically by the same submitted form, since they are generated from the common template, they have the similarities in their structures, thus the content differences can be discovered quickly by the fuzzy match; and by the intelligent identification to the page elements, the Agent can find the changes of the pages under testing, thus it can adjust the testing scheme accordingly.

4.3 Monitoring the Testing Process

The testing monitor takes charge of the control and harmony with the whole testing tasks, and is generally taken on by testers or some intelligent entities. In order to improve the intelligent degree of the testing, liberate the testers drastically and facilitate the communication and interaction with the test-execution Agent, we also select the intelligent Agent as the testing monitor so as to own the consistent in the presentation and contact.

Combined with Figure 1 and Figure 3, we know the testing monitor is the hinge of the whole testing process, which is associated with the input and output of the testing, and also monitors the running states of each test-execution Agent. The workflow of the test-monitor Agent is described as follows:

(1) After the test cases are send to the test-monitor Agent as the input information, the Agent estimates the scale of the testing tasks so as to determine the needed number of the test-execution Agents, and divides the tasks into several cells for each test-execution Agent.

(2) Based on the running states of each test-execution Agent, the test-monitor Agent carries out the real-time management and control, so as to ensure the load equilibriums and the collaborative communications among the test-execution Agents.

(3) When the exceptions come forth during the testing process, the test-monitor Agent adopts some schemes to deal with these problems, excluding the faults and eliminating the influences in time.

The realization of the above functions requires a great deal of trainings for the test-monitor Agent beforehand, thus generate some disciplinarians and regulations, and store them into the repository of the test-monitor Agent; at the same time, utilizing the self-learning property of the Agent, it can improve the intelligent degree of itself ceaselessly until it has the ability to solve the exceptions independently and availably.

5 CONCLUSIONS AND FUTURE DIRECTIONS

Web applications develop very rapidly, but their properties, such as distributed, dynamic, multi-platform and interactive along with the heterogeneous and autonomous running environments, make it much difficult to carry out the testing. And the testing process usually needs a great deal of testing cost, however, the testing efficiency is not high. So it is urgent to adopt some testing assistant tools to improve the automatic level of the testing process; at the same time, it is also demanded that these tools should have certain degree of intelligence to fit for Web applications' dynamic and evolving characters.

Based on the previous work, this paper established an executing scheme for the Web application testing, and the realizing techniques have certain degree of automatization and intelligence. By introducing the related work, we explained the necessity and feasibility of carrying out the Web application testing automatically and intelligently. Then, the executing process of the Web application testing were detailedly described and thoroughly analyzed, and in this way, we determined the steps and flows of the testing execution along with the related contents such as the adopted techniques and tools. Next part was concerned with the realization techniques. In this section, we improved the capture-replay technique and made it suitable for the dynamic characters of Web applications, and adopted the intelligent Agent to realize the precise testing behaviours, correct replaying, and the monitor, management and exception-handler of the whole process for the Web application testing.

Future directions are focused on the research of the realizing techniques of the testing process for Web applications. Since the testing process is divided into three phrases, i.e. the prophase preparation, testing execution, and comparing testing results, and the relationships among them are close and influencing each other. So it is urgent to deeply research the details of each separate phase firstly, then integrate them into a related whole, so as to optimize and consummate the testing process and equip them with better pertinence and adaptability. When the realizing details are referred, we should pay more attention to the developing trends of Web applications, and furthermore select some valuable assistant tools, mathematic theories and realizing techniques to the testing process of Web applications, so as to further improve the automatic and intelligent level of the testing. And the future work still includes the research and development of the related tools.

REFERENCES

Anupam, V., Freire, J., Kumar, B., and Lieuwen, D. , 2000. Automating Web Navigation with the WebVCR, *Proc. of WWW*, pp. 503-517.

Belkin, N.J., and Croft, W.B., 1992. Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM*, 35(10): 29-38.

Gao, J., Chen, C., Toyoshima, Y., and Leung, D., 1999. Engineering on the Internet for Global Software Production, *IEEE Computer*, 32(5): 38-47.

Guttman, R.H., Moukas, A.G., and Maes, P., 1998. Agent-mediated Electronic Commerce: A Survey, *Knowledge Engineering Review*, 13(2): 143-152.

Hewitt, C., 1977. Viewing Control Structures as Patterns of Passing Messages, *Artificial Intelligence*, 8(3): 323-364.

Kallepalli, C., and Tian, J., 2001. Measuring and Modeling Usage and Reliability for Statistical Web Testing, *IEEE Trans Software Engineering*, 27(11): 1023-1036.

Krulwich, B., 1997. Automating the Internet: Agents as User Surrogates, *IEEE Internet Computing*, 1(4): 34-38.

Kung, D.C., Liu, C.H., and Hsia, P., 2000. An Object-Oriented Web Test Model for Testing Web Applications, *Proc. of the Asia-Pacific Conference on Quality Software (APAQS)*, pp. 111-121.

Liu, C.H., 2002. A Formal Object-Oriented Test Model for Testing Web Applications, Doctor Dissertation.

Murugesan, S., 1998. Intelligent Agents on the Internet and Web, *IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control*, 1(1): 97-102.

Ricca, F., and Tonella, P., 2000. Web Site Analysis: Structure and Evolution, *Proc. of International Conference on Software Maintenance (ICSM'2000)*, pp. 76-86.

Warren, P., Boldyreff, C., and Munro, M., 1999. The Evolution of Websites, *Proc. of the Int. Workshop on Program Comprehension*, pp. 178-185.

Wooldridge, J., and Jennings, N.R., 1995. Intelligent Agents: Theory and Practice, *Knowledge Engineering Review*, 10(2): 115-152.

Xu, L., and Xu, B.W., 2003. Applying Users' Actions Obtaining Methods into Web Performance Testing, *Journal of Software (in Chinese)*, 14(Suppl.): 115-120.

Xu, L., Xu, B.W., Chen, H.W., Chu, W., Lin, J.M., and Yang, H.J., 2003. Test Web Applications Based on Agent, *Journal of Software*, 14(Suppl.): 9-16.

Xu, L., Xu, B.W., Nie, C.H., Chen, H.W., and Yang, H.J., 2003. A Browser Compatibility Testing Method Based on Combinatorial Testing, *Proc. of the International Conference on Web Engineering (ICWE)*, pp. 310-313.