http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 6 (1): 74-81, 2007
ISSN 1812-5638

A Strategy for Testing of Web Based Software

[1]Ghulam Mustafa, [2]Abad Ali Shah, [1]Khadim Hussain Asif and [1]Amjad Ali
[1]Department of Computer Science and Engineering, UET Lahore, Pakistan
[2]R and D Center of Computer Science, UET Lahore, Pakistan

Abstract: The computing environment for web applications is dynamic and complex as internet is heterogeneous, distributed, multi-platform, multimedia, multilingual and cooperative wide area network. The web software itself is also dynamic and rapidly evolving. This complex and dynamic computing environment, complex user requirements, new features and constraints make the testing of web-based software a challenging task. Traditional testing techniques are not sufficient because they do not address the challenges associated with the web-based applications. This study reviews the techniques for the testing of web based applications and gives a strategy to use the testing techniques meeting the challenges of testing web-based software.

Key words: Web, testing, software engineering, software quality

INTRODUCTION

Internet is rapidly creeping into all sectors of our life and has become a vast computing environment. The computing environment is complex as internet is heterogeneous, distributed, transparent, multi-platform, multilingual, multimedia, autonomous and cooperative wide area network. The environment is also changing with time. The software that use internet as computer face not only this complex and dynamic computing environment but also involves complex user requirements, constraints and new features. The software itself is also dynamic and rapidly evolving. This makes testing of web-based software a complex task. Traditional testing techniques are not adequate for web based applications because they do not address the problems associated with the computing environment, new features and constraints of web-based applications. At present, web-based software testing relies on borrowed testing techniques. A little work has been done on testing web applications (WAs) and needs research to explore this area.

Tremendous effort has been made to assure the quality of traditional programs, resulting in variety of testing techniques for both stand-alone and distributed systems. Most work on web applications has been on making them more powerful and relatively less attention has been given to ensure their quality. The main techniques for testing web applications are applying the traditional testing methods because web applications belong to the software and have some commonness with the traditional software. Web applications share some characteristics of client-server and distributed applications and some of these techniques can also be used to help assure the quality of web applications. However there are a number of special aspects of web applications that prevent them from being directly used. It needs either modification to existing or completely new techniques. Some of the distinguishing features of web applications compared to traditional and distributive applications are given below:

Role of client and server: In client-server system the role of the clients and servers and their interactions are both predefined and static compared to web applications where client side programs and contents may be generated dynamically.

Dynamic environment: The environment for web applications is not predefined and is changing dynamically i.e., hardware and software are changing, configuration are ever-changing, the heterogeneity of the hardware and software components, the extremely loose coupling and dynamic integration. Web applications often are affected by these factors that may cause incompatibility and interoperability issues.

Execution control: For traditional programs, the control flow is fully managed by the program, so the user cannot affect it. When executing web application, users can break the normal control flow without alerting the program controller. For example, users can press refresh button in the web browser, which totally changes the execution context, causing unexpected results.

Corresponding Author: Ghulam Mustafa, Department of Computer Science and Engineering, UET Lahore, Pakistan

Faster maintenance: Web applications also have much faster maintenance requirements than most traditional software. Therefore maintenance not only needs to be done more frequently, but also needs to be done more efficiently due to the peculiar time-to-market pressure for web applications.

Diverse user population: Web application users are diverse, huge population size with different educational backgrounds, age, gender, cultures, aesthetic sense and geographical locations resulting a very complex user profile. The load on web access due to this diversity of population is not predictable.

Content driven: Web applications use hyper media to present text, audio, video and graphics to user.

Other features: The web hosts an enormous range of technologies, including links, downloaded programs and dynamic content and stateless aspects each with its own testing challenges.

We need to design quality web sites that are not only attractive, user friendly and well-organized, but also easy to navigate and fully-functional. A web site should possess all of these characteristics to be successful. A visually attractive web site soon loses its appeal when important information is not easily accessible and navigation is confusing. So thorough testing of the web software is necessary to eliminate problems which seriously detract user from a web site. This requires huge additional testing requirements like dead-ends, dead links, missing branches, unreachable items, cycles, spelling mistakes, grammatical errors, browser incompatibilities, consumption of un-produced data, web page functionality, plug-ins like shock wave, real audio, performance testing over various connections to determine speed, graphics loading etc. However thorough testing of web applications is not possible due to a number of constraints.

Constraints in web applications testing: We analyze some challenges faced in testing web applications that make thorough testing of web applications a mission impossible:

Immediacy: The style of development of web applications is of Rapid Application Development (RAD) type. As it requires shorter developing and testing time so it is not possible to test every aspect of the web application. Challenges for a web tester are what to test and when to stop testing and optimize resources to get most from the testing effort.

Importance of quality attributes: For traditional programs, correctness and efficiency are usually the most important quality factors. Web applications are usually information focused and other quality features like usability and reliability are often more important depending upon the application category of the web software. Compatibility and interoperability are also urgent requirements and may cause problems that are more serious than with traditional programs. The requirements may also be conflicting. In addition the quality metrics are not available. This is another challenge for a Web applications testing team.

Lack of documentation: If the analysis and design documents are available, the testing requirements can be determined more conveniently. However, the Web applications are generally short of the developing documents. So, in order to analyze and present the testing requirements, firstly we have to determine the testing targets and then describe the testing objectives.

Testing objectives: One of the key challenges in web application testing is to define testing objectives. There is variety of application categories making it difficult to define testing objectives. A few tasks associated with defining the testing objectives have been investigated that are identifying threats to customer satisfaction, understanding the risks of putting the software into production, product's behavior against a standard and minimizing technical support costs. However the story is not so simple.

Retesting and test automation: An important aspect of web applications testing is its retesting requirement and test automation. For many good reasons, given below, re-testing of web applications is a necessity in WAs life cycle.

- It necessary to retest WAs regularly during its life cycle due to the evolutionary nature of the Web technologies and applications.
- Same test is required to be repeated on many browsers.
- Web applications can be updated quickly and software developed quickly is often bug-ridden unless thoroughly tested. This requires automatability.
- There is no brand name loyalty for web. About 40% customers do not return to web site if they encounter service failure (Patel, 2001). Certain performance parameters are vital to retain customers on the web. Load is such a parameter. Overloaded servers often refuse service. Such testing is necessary to

guarantee the failure free service to the customer. This demanding nature of WAs shows that automated testing is an important part of website testing. In general a tester should automate where appropriate however regression tests should be automated to the furthest extent possible.

Cost: Shrinking budgets and high cost of web testing makes it difficult to test every aspect of the web application. Optimization of resources to get most from the testing is a challenge for web tester.

Problems trying to test: We face number of problems while trying to test. The problems that can occur depend on the application, technologies used, settings and environment, user software configuration etc. Few common problems that can arise with web applications are:

- Testing for session and cookies require simulating a browser that requires long time to cause timeouts.
- Exhaustive test of entries in a given form submission is impossible.
- The site may have to go down while being tested.
- Testing on a simulated test site may not provide the same environment.
- Stress testing is hard to perform on a shared hosting environment while the site is live. Automated testing usually populates the database heavily with generated test cases that not only need to be cleared out, but also put additional stress on a live site. It may also cause serious problem to consistency of data.

Special attributes: Web applications also have features that are not present in client-server and distributed systems. These include session control, cookies and the stateless aspect of HTTP. Therefore, new solutions are necessary for these special features.

Security: Security of web applications is another challenge for web applications developers. Web applications execute in an environment where varieties of cooperating hardware and software technologies are involved. There is large number of security aspects of web applications. Not only careful planning of security at design time is hard but also testing and maintaining security of web applications is a challenge for the testing team.

The challenges given above make testing of a web application a difficult task. The current testing field focus is on techniques for unit-level testing and sub-system testing. However web applications testing lack systematic testing methods and techniques to cope with the challenges of testing. Currently the common strategies and techniques for web testing are adopted and usually driven by specific tools, metrics or immediacy, rather than objectives to achieve. They do not yet provide much help to test web applications based on the constraints as described earlier. The key challenge in Web testing lies in adoption of a testing strategy and the allocation of resources behind it to support test execution (Hung, 2004).

As most of the web based applications belong to business world and must be treated in the same context. There are a number of constraints as discussed above in the development and testing of web applications. Dealing with these constraints and finding an optimal solution to the problem of testing web applications is the main focus of this research. We use the principle of economics to define testing strategy i.e., satisfying unlimited wants using the limited resources because it is best suited to our web application development problem i.e., Unlimited Testing Requirements using Limited Resources. In this study we give an economics based testing strategy to fulfill the testing challenges of web applications.

SURVEY OF WORK RELATED TO WEB APPLICATIONS TESTING

A large number of researchers have worked in the field of web application testing. Each researcher has focused in certain test area based on architecture, model, attributes, scenario, domain functionality and other nonfunctional qualities. Testing of web applications using white box testing techniques based on two layers architecture is given in (Tonella and Ricca, 2004). A hierarchical strategy for testing web based applications and assuring their reliability based on three-tier architecture of web applications is given in (Tian *et al*., 2003). An N-tier architecture for testing web applications is given in (Xiong and Robert, 2003) that claims to support much greater application complexity, higher traffic and stronger site security by separating presentation from business logic. WAs involves many types of testing, such as GUI testing, functional testing, database testing etc. A multi-approach testing methodology for WAs has been suggested in (Xu *et al*., 2005) which applies multiple models, implementation, user profiles and multiple test methods such as grey box, black box, white box. It chooses different tools to test different aspects of WAs and focuses on unit and integration levels according to the 3-tier architecture of WAs. A number of techniques based on model have been reported in literature. The

models employed Include analysis model, such as use case, object state diagram; and design model, such as navigation model, business model, data model, sequence diagrams, event model. A testing technique based on analysis model is given in (Ricca and Tonella, 2001). Functional defects eliminating through model- based testing is given in (Becker, 2005). Model-based testing is useful and effective but expensive and has limitations (Bertolino *et al*., 2005). An anti-model testing philosophy and approach has been discussed in (Bertilino *et al*., 2004). Some issued, challenges and solutions of testing component based software are given in (Wu and Gao, 2004). A recent work (Xu *et al*., 2005) on web application testing that give a model to test based on their specialty. A methodology to test for time critical web application employing rapid WAs development process is given in (Patel, 2001). A strategy to test web applications is given in (Duchnowsky, 2000) that focusing usability and few other challenges to test web applications. A very useful and informative source for web application testing (Stout, 2001) that discusses the best practices in testing web based software. A risk based testing strategy based on these best practices is given in (Gerrard and Thompson, 2002). Risk based testing of web applications has been considered by many researchers. We refer to the work of (Besson, 2005) and (Gerrard and Thompson, 2002). Methods based on scenarios are mature since they have been applied and validated over the past several years, but the development of attribute model-based architecture evaluation methods is still ongoing. Most of the approaches discussed in literature are Model-based that are certainly useful and effective but has limitations. A detailed discussion on these limitations is given in (Rosenblum, 1998). As most of the web applications are component based so we are not using model base testing of the web due to above given reasons.

We are moving towards transparent computational environment offered by internet. In this situation static testing and black box testing techniques for web related problems are more important as compared to techniques based on the architectural models.

The current testing focus is on unit-level testing and sub-system testing. Systematic testing methods and techniques, quality metrics and dependability of web-based applications are still unexplored areas of WAs testing. The common strategies and techniques for web testing are adopted and usually driven by specific tools, metrics or immediacy, rather than objectives to achieve. They do not yet provide much help for the multi-system testing needed for applications on the web. The key challenge in Web testing lies in adoption of a Web testing strategy and the allocation of resources behind it to support execution. In this study we are presenting testing strategy that prioritizes test cases based on the constraints and challenges. There are number of constraints as discussed earlier in the development and testing of web applications. Facing these constraints and finding an optimal solution to the problem of testing web applications based on the schedule, objectives and economic feasibility is the main focus of this research. We use the principle of economics 'satisfying unlimited wants using the limited resources' to meet the challenges of our web application testing. We give a solution to the problem 'Satisfying Unlimited Testing Requirements using Limited Resources.'

PROPOSED WEB APPLICATIONS TESTING STRATEGY

Most of the approaches discussed in literature are Model-based that are certainly useful and effective but has limitations. For example:

- Model-based testing is inherently complex that requires a deep expertise in formal methods even where tool support is available.
- Difficulty in designing test cases that force the implementation to take a defined path as identified in the model derived test sequences.
- Component-based software development, a system is generally obtained by assembling already existing components or legacy systems, for which a specification or the source code may not be available. In such cases, model-based testing is not applicable.

Although most of the web applications are component based however we are not using model base testing of the web due to above given reasons. As most of the web based applications belong to business world and must be treated in the same context. We want thorough testing of the web applications we build. This helps us to eliminate problems which seriously detract from a web site: dead-ends, dead links, spelling mistakes, grammatical errors, browser incompatibilities.

There are number of constraints as discussed above in the development and testing of web applications. Facing these constraints and finding an optimal solution to the problem of testing web applications based on the economic feasibility, is the main focus of this research. We use the principle of economics satisfying unlimited wants using the limited resources to meet the challenges of our web application testing. We give a solution to the problem Satisfying Unlimited Testing Requirements using Limited Resources. We base our

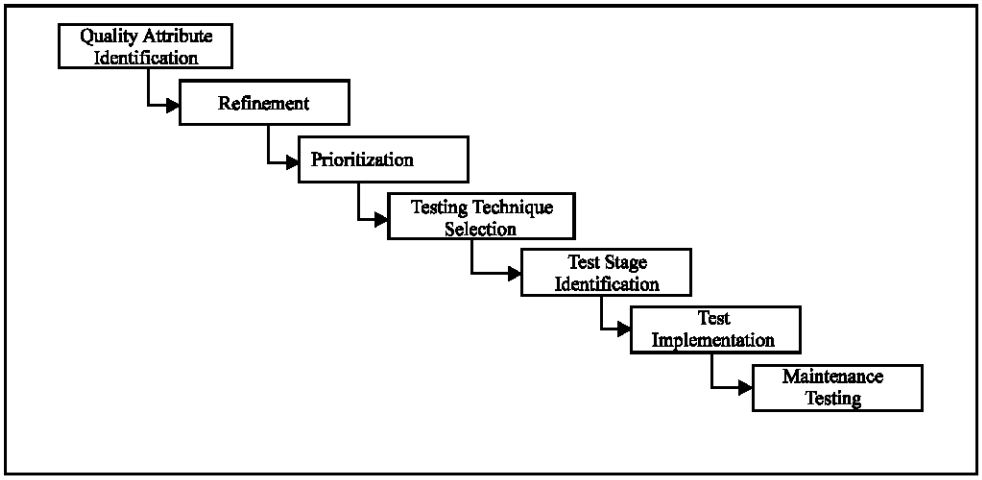

Fig. 1: Prioritization of attributes and test stages identification process

strategy on the principle Start early, test often, end never. To achieve the testing goal we suggest following steps as shown in Fig. 1.

Step 1: Quality attribute identification: There are number of quality characteristics that are important for a web application depending on the requirements of the application domain. In literature there are number of method and attributes associated with each quality characteristics. A detail survey of techniques and attributes require research in its own capacity. There are number of sources of information to gather the test requirements for web base applications. Some of these include defined business requirements and success criteria, software application category, project schedules, requirements definition, design specification, operability standards, customer acceptance criteria and customer acceptance test specification. The problems which seriously detract from a web site like dead-ends, dead links, spelling mistakes, grammatical errors, browser incompatibilities need serious considerations. Using these information sources we can define the objective of the testing and testing requirements.

Step 2: Atomic quality attributes: After identification of the key quality attributes, we refine quality attributes to atomic level to identify the test cases requirement. For each of the quality attribute there are number of associated problems, measurable and metrics. These are listed against each quality attribute. Each atomic test attribute correspond to a number of test cases. These test cases are then prioritized based on the economic criteria and defined goals. We analyze test cases corresponding to each atomic test attributes as follows:

- Eliminate all test cases that have no concern with the quality of the particular project under development.
- Identify issues that are absolutely necessary for the project considering time, cost and risk factor.
- Prioritize the remaining issues.

Step 3: Prioritize the atomic test attribute: We suggest four levels priority and label them in terms of decreasing priority:

Smoke priorities (0): This priority is assigned to the test cases that are absolutely important to the software. Time and cost are important factors to decide this priority. If there is severe time and cost pressure to perform testing stick to smoke testing.

Need/Normal priority (1): This priority is assigned to those attributes that are necessary for usability, dependability and performance.

Comfort/Expectation priority (2): This priority is assigned to those attributes that reduce risks of software failure without increasing much time and cost penalty.

Luxury/Excitation priority (3): This priority is assigned to those attributes that minimize risks of software failure and need time and cost.

These are the terms taken from economics as the basic idea on which the priority is based is also taken from the theory of economics.

Setting the priority

- Identify business-critical attributes
- Assign weight to each of the selected attributes based on

- Defined objectives
- Time to complete test
- Cost of resources to conduct the test
- Risks
- Test priority from highest risk areas => lowest risk areas
- Conduct a periodic risk assessment
- Assign weights corresponding to the level of priority.

Step 4: Selection of testing technique: At this stage we identify the testing techniques for each of the selected test cases. Priority of selection of testing technique to test the issue is made as follows:

- The testing techniques that support test first programming should have higher priority
- The techniques that are automat able should be given higher priority.
- In addition to this the factors like time, cost, trained manpower availability and test accuracy and test versatility are also the deciding parameters.

For lower priority attributes select a testing technique that is relatively more economical.

List each of the selected testing technique against the test attribute.

There are number of different classification of testing techniques in literature like Static, Dynamic, Functional, Non-Functional, unit, Integration, black box, white box, statistical, regression etc. For our purpose we are following the classification of techniques as given in (Shah, 2003) and (Gerrard and Thompson, 2002). We group the Web testing techniques into five classes based on the nature of testing:

- Static testing techniques
- Infrastructure testing techniques
- Functional testing techniques
- Non functional testing techniques
- Integration testing techniques.

Each of the testing techniques may fall under one or more of the above given categories. In addition our quality issues may require one or more of these techniques. A few techniques belonging to each class suggested as best practices in literature (Stout, 2001) are selected for demonstration.

Step 5: Determine testing stage: To make the testing activity model and architecture free we classify our testing into five technical groups. These stages do not always fit into the traditional stages. This grouping is similar to the suggested in (Gerrard and Thompson, 2002) (Fig. 2).

Desktop testing: This is testing in desktop environment using browser and testing simulators.

Subsystem testing: It is testing of subsystems in the real environment. It is integration testing of server based components with the web page.

System integration testing: It is integration of all components of the system in isolation from other systems.

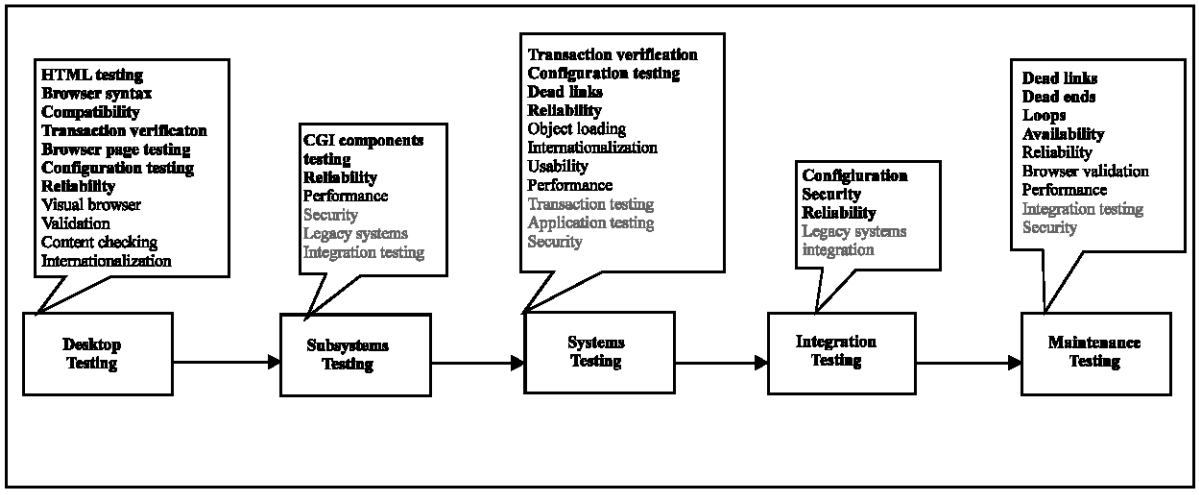

Fig. 2: Prioritized attributes and implementation stage

Large scale integration testing: It is integration of the system with legacy and other systems.

Monitoring stage: We monitor the web site by testing the some important attributes of web by retaining automated tests.

Step 6: Test implementation:

- Generate test data corresponding to the prioritized test cases.
- Use Test-first programming approach i.e. start testing before development even commences. Allocate more resources for such testing as it saves even more time and money on debugging and fault correction. The better to integrate testing, design and implementation testing into process.

Step 7: Maintenance testing: It is important to monitor the web because users of web site are not bound to report service failure or degradation. Web site monitoring involves the same activities as testing in development cycle. The most important testing characteristic required is availability, link checking, load checking and security. Maintenance testing needs automatability of testing. Where it is not possible manual testing should be done on reasonable interval of time.

CONCLUSIONS

This study presented our work in web application testing, which includes the requirement analysis, requirement prioritization, testing techniques selection, testing techniques prioritization, test case generation and, testing execution. Main emphasis is to prioritize the test cases at atomic level. Testing priority is based on the constraints, economic gain and loss. Test cases are designed based on the priority.

An immediate challenge in implementation of this technique is the collection of atomic attributes and related metrics. In literature there are number of methods and attributes associated with each quality characteristics. A detailed survey of techniques and attributes requires research in its own capacity. Our immediate future work will focus on the search of comprehensive set of test attributes and associated metrics. A reasonable approach for cost of testing and failure risk and associated cost analysis is required for each of the attribute based on the category of web application.

ACKNOWLEDGMENT

The authors acknowledge the financial support and enabling role of HEC Islamabad, PAEC Islamabad and UET Lahore, Pakistan through Development of S and T Manpower through Indigenous PhD (300 Scholars) scholarships scheme.

REFERENCES

Becker, P., 2005, Eliminating functional Defects through Model-Based Testing, Retrieved on 14-4-2005, http://www.stickyminds.com/.

Besson, S., 2005. A strategy for risk-based testing, software Qual. Eng. http://www.sqe.com/.

Bertolino, A., E. Marchetti and H. Muccini, 2005. Introducing a reasonably complete and coherent approach for model-based testing. electronic notes of Theor. Computer Sci., 16: 85-97.

Bertilino *et al*., 2004. Towards anti-model-based testing. Fast Abstract in The International Conference on Dependable Systems and Networks (DSN 2004), Florence.

Duchnowsky, P., 2000. Building quality into software-meeting the challenges of testing and usability. International Conference on Practical Software Quality Techniques (PSQT'2000), Minneapolis.

Gerrard, P. and N. Thompson, 2002. Risk-based E-business testing. Artech House.

Hung, Q.N., 2004. Web application testing beyond tactics, proceedings of the 6th IEEE International Workshop on Web Site Evolution (WSE'04), pp: 1550-1564.

Patel, E., 2001. Rapid SQA: Web testing at the speed of the internet, international conference on practical software quality Techniques and Testing Techniques (PSQT/PSTT'2001 East), Florida.

Ricca, F. and P. Tonella, 2001. Building quality into web applications: Analysis and Testing of Web Applications, IEEE 0-7695-1050-7/01.

Rosenblum, D.S., 1998. Challenges in exploiting architectural models for software testing, proc. International Workshop on the Role of Software Architecture in Testing and Analysis.

Shah, B., 2003. Testing maturity model (TMM) Framework for Internet Based Applications, Online Proceedings of the Asia SEPG (SM) Conference' 2003, available at: http://www.qaiindia.com/Conferences/presented/bharat_lms.pdf.

Stout, A.G., 2001. Testing Websites: Best Practices, Revere Group. Retrieved from web on Feb 12, 2005, www.reveregroup.com.

Tian, J., L. Ma, Z. Li and A.G. Koru, 2003. A hierarchical strategy for testing web-based applications and ensuring their reliability. 27th IEEE Annual International Computer Software and Applications Conference, COMPSAC 2003, pp: 702.

Tonella, P. and F. Ricca, 2004. A 2-Layer model for the white-box testing of web applications, Sixth IEEE International Workshop on Web Site Evolution (WSE'04), pp: 11-19.

Wu, Y. and J.Z. Gao, 2004. Testing component-based software- issues, challenges and solutions. Third International Conference on COTS bases Software systems, ICCBSS 2004, Redondo Beach, USA.

Xiong, P. and L.P. Robert, 2003. A multi-approach testing methodology in ttcn for web applications. Chillarege Press Fast Abstract ISSRE.

Xu, L., B. Xu and J. Jiang, 2005. Testing web applications focusing on their specialties. ACM SIGSOFT Software Eng. Notes, 30: 10.