Information Systems 43 (2014) 20–54

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

# Two decades of Web application testing—A survey of recent advances

Yuan-Fang Li *, Paramjit K. Das, David L. Dowe

*Clayton School of Information Technology, Faculty of Information Technology, Monash University, Clayton, VIC 3800, Australia*

ARTICLE INFO

*Article history:*
Received 1 February 2013
Received in revised form
21 December 2013
Accepted 1 February 2014
Recommended by: F. Carino Jr.
Available online 14 February 2014

*Keywords:*
Software testing
Web applications
World Wide Web
Web testing
Survey

ABSTRACT

Since its inception of just over two decades ago, the World Wide Web has become a truly ubiquitous and transformative force in our life, with millions of Web applications serving billions of Web pages daily. Through a number of evolutions, Web applications have become interactive, dynamic and asynchronous. The Web's ubiquity and our reliance on it have made it imperative to ensure the quality, security and correctness of Web applications. Testing is a widely used technique for validating Web applications. It is also a long-standing, active and diverse research area. In this paper, we present a broad survey of recent Web testing advances and discuss their goals, targets, techniques employed, inputs/outputs and stopping criteria.



**Contents**



* Corresponding author.
*E-mail addresses:* yuanfang.li@monash.edu (Y.-F. Li), paramjit.das@monash.edu (P.K. Das), david.dowe@monash.edu (D.L. Dowe).

http://dx.doi.org/10.1016/j.is.2014.02.001
0306-4379 

## 1. Introduction

Software testing [1] has been widely used in the industry as a quality assurance technique for the various artifacts in a software project, including the specification, the design, and source code. As software becomes more important and complex, defects in software can have a significant impact to users and vendors. Therefore, the importance of planning, especially planning through testing, cannot be underestimated [1]. In fact, software testing is such a critical part of the entire process of producing high-quality software that an industry may devote as much as 40% of its time on testing to assure the quality of the software produced.

In software testing, a suite of test cases is designed to test the overall functionality of the software—whether it conforms to the specification document or exposes faults in the software (e.g., functionality or security faults). However, contrary to the preconceived notion that software testing is used to demonstrate the absence of errors, testing is usually the process of finding as many errors as possible and thus improving assurance of the reliability and the quality of the software [1]. This is because, in order to demonstrate the *absence* of errors in software, we would have to test for all possible permutations for a given set of inputs. However, realistically, it is not possible to test for all the permutations of a given set of input(s) for a given program, even for a trivial program. For any non-trivial software systems, such an exhaustive testing approach is essentially technologically and economically infeasible [1]. The main objectives of any testing technique (or test suite) can be summarised as:

- Testing is carried out mainly to demonstrate the presence of errors that exist during a program execution.
- A good testing technique will have a higher chance of discovering an error.
- A successful test case should discover a new fault or a regression fault.

Ever since the creation of the World Wide Web in the early 1990s [2], there has been a tremendous increase in the usage of Web applications in our daily lives. The idea behind the World Wide Web was possibly envisioned by C. S. Wallace as early as 1966 [3, pp. 244–245], where he envisioned that a central computing system (or the server), or a bank of computers, could be used to carry out various computing tasks, such as paying bills, ordering goods, carrying out engineering tasks, etc., for a large number of users. In these instances, the time required would be shared equally amongst all users, which would make the process economically feasible. This concept was appropriately labelled "Time-Sharing", since the time would be shared amongst all users.

A *Web application* is a system which typically is composed of a database (or the back-end) and Web pages (the front-end), with which users interact over a network using a browser. A Web application can be of two types – *static*, in which the contents of the Web page do not change regardless of the user input; and *dynamic*, in which the contents of the Web page may change depending on the user input, user interactions, sequences of user interactions, etc.

The profound transformative impact the Web and Web applications have brought about on our society has long been acknowledged. Somewhat surprisingly, however, there seems to be very limited research that has been done in surveying the different recent advancements made in the field of Web application testing over the past 20 years. To the best of our knowledge, the only other surveys in this field consists of an early review by Di Lucca and Fasolino [4] on general approaches to Web application testing, and a survey by Alalfi et al. [5] focussed on the modelling aspects of Web application testing. Therefore, this survey paper provides a much needed source of detailed information on the progress made in and the current state of Web application testing.

Compared to traditional desktop applications, Web applications are unique in a number of ways, and such uniqueness presents new challenges for their quality assurance and testing.

- Firstly, Web applications are typically multilingual. A Web application usually consists of a server-side backend and a client-facing frontend, and these two components are usually implemented in different programming languages. Moreover, the frontend is also typically implemented with a mix of markup, presentation and programming languages such as HTML, Cascading Style Sheets (CSS) and JavaScript. The presence of multiple languages in a Web application poses additional challenges for fully automated *continuous integration* (CI) practices, as test drivers for different languages need to be integrated into the CI process and managed coherently.