Abstract

Performing an effective system test requires an in-depth knowledge of the domain and system being tested and an effective test design method, coupled with the right test infrastructure. The current state-of-the-art does not respect the constraint that Subject Matter Experts (SMEs) are in short supply, nor exploit the fact that similar systems have been built and tested in the past. Most current state-of-the-art methods do not leverage the various models and specifications and other design or implementation artifacts to simplify the test process. While there is a lot of focus on generating test inputs from specifications, inadequate attention has been paid to verifying test results. This paper tries to highlight some of the problems and research opportunities in this area.

1. Introduction

A system test is meant to exercise the complete system in a manner a user would exercise it. System testing is a key activity in the software life cycle. It is a part of the core service offerings from most testing service providers. There are various challenges in the way system testing is carried out.

2. Opportunities/Problems

Let us consider system testing from various angles.

2.1. Effectiveness. System testing is very much a manual, hence effort-intensive process and is often done in an ad-hoc manner. It is done without an explicit focus on test case design. Tests are usually specified in a verbose manner using a natural language. The test data is either assumed or is informally generated and is a part of the test specifications. There is no separation of concerns as far as the test designer and test executor are concerned. This affects the overall effectiveness of the whole process.

2.2. Complexity. Test automation adds further complexity. Test cases end up being maintained in the form of executable test scripts. The whole process then becomes bottom-up and even more complex with changing application functionality. Various aspects like test design, test scripting, test data generation, test execution, and verification are often maintained separately, while references are maintained manually. Creation of test data is often done in an ad-hoc manner, hence it is difficult to maintain; moreover, any sort of change requires repetition of the test data creating and setup steps.

2.3. Reuse. Acquired domain knowledge in terms of models or specifications is not reused, even for testing of packaged software components. Reuse of test-ware like test cases, test scripts, test data is seldom done.

2.4. Verification. Verification is often a taken-for-granted thing. It is often assumed and unstated. Not much help is available on this front from the research community.

3. Conclusion

A top-down integrated approach that explicitly addresses test design, leverages test data generation, emphasizes verification, simplifies test representation and promotes reuse of domain knowledge and test-ware is needed to make system testing more effective.

4. References