VIEW POINT

# Five Commandments for Successful COTS Package Testing


## Abstract

Ineffective COTS implementation will cost you

Adopting commercial off-the-shelf (COTS) products or packages like ERP, CRM, and HR management systems to fulfil a range of enterprise functions is a crucial decision involving huge investment.

Most implementations do not identify testing as an independent function required during the implementation of the COTS product, while others do not engage testing teams early enough. Only a minority have a successful implementation without any testing-related challenges. The rest discover that the eventual cost of implementation and on-going maintenance have dented their overall business case.


Infosys
be more

Some worrying trends common to such failed implementations are:

- Expensive functional consultants conduct testing
- Business users' involvement is higher than required for complementing testing
- Lack of sufficient testing coverage during the testing phase leading to defects showing up in UAT or, even worse, in the production environment
- Testing is not recognized during on-going maintenance

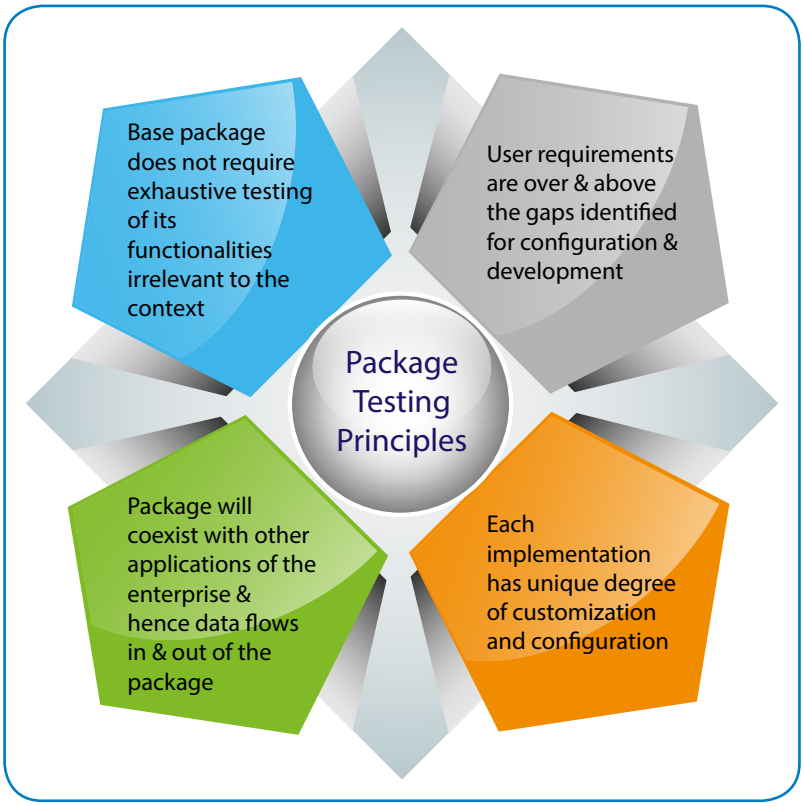
This point of view elaborates the rationale of independent testing for package implementation/upgrades and highlights arguments for the benefit of the yet undecided.

External Document © 2017 Infosys Limited

## What is your strategy for COTS/Package Testing?

The complexity and risk associated with implementation has a direct correlation with the scope expected to be realized from the package. Companies are increasingly realising the role independent testing teams with specialized testing skills can play in overcoming the risks specified above.

The package testing dogma during implementation/on-going maintenance of a package can be laid out as follows:




- Testing is not required for the base package, at least for the irrelevant conditions/functionalities.
  - For instance, for an order creation and printing application created in Microsoft Excel, it is not required to test the MS Excel product core functionalities like 'Print Preview' and 'Print Properties' options.
- The gaps identified for configuration/ development are different and are a subset of what users require.
  - Taking the same example of MS Excel, for a requirement of the ability to always print the sales orders in the landscape mode, the MS Excel core functionality of printing is a subset of the overall workflow. It must be tested in conjunction with the configuration changes in the order creation and printing application.
- Any package implementation/ operation will involve a degree of configuration and code development.
  - All products/packages should be configured before use and customized to address user requirements. Every package implementation is unique as a result of the degree to which the product is customized and configured.
- The package will co-exist with other applications/packages and data will flow into or from the package.
  - For example, an online retailer may use an order fulfilment package to fulfil the order received. To share the shipping information with the carrier, this package has to be integrated with the delivery systems of the carrier. Testing of the functionality for the changes is a job half done.
  - For externally facing applications developed on COTS (Oracle ATG, Salesforce.com, etc.) emphasis of performance and security requirements are self-explanatory and rightly justified.

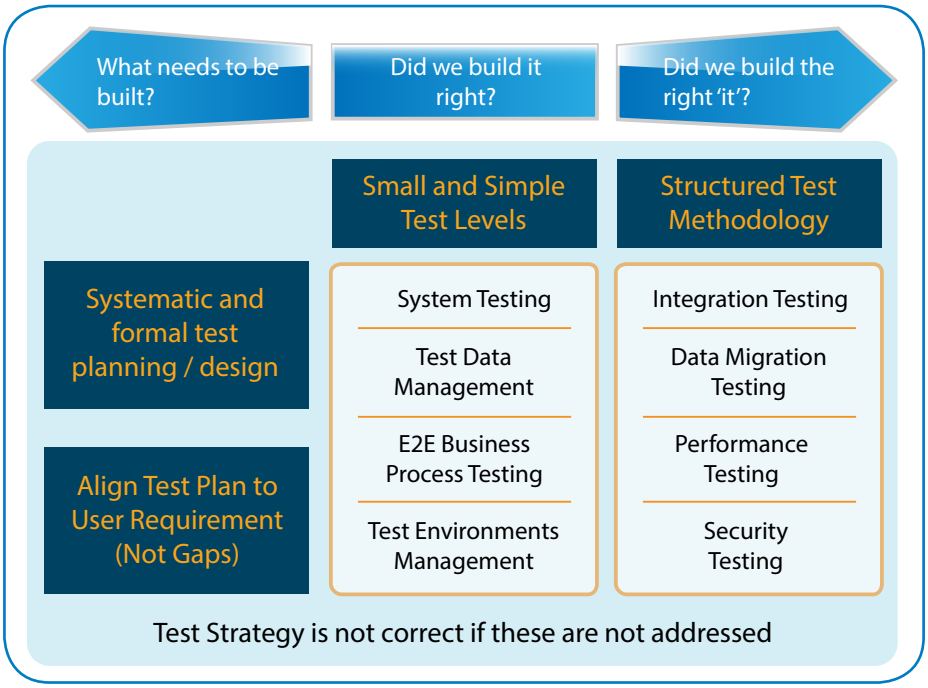
When the COTS product is primarily implemented for back-office systems (Oracle PeopleSoft, Sterling Commerce, etc.), special considerations for improved performance and data security are critical inputs. In addition, data retention and disaster recovery are equally important requirements when compared to the functional expectations.

This calls for specialist testers, who prioritize based on the identified risks, appreciate the changes from a user perspective and accordingly implement the right set of strategies.

External Document © 2017 Infosys Limited

During the early stages, when different components of the products are built or configured and put together, it is important to answer the question: "Did we build it right?" Effective testing at this stage tends to be more technical in nature. As the package reaches finishing stages, it is important to answer the question: "Did we build the right thing?" Testing at this stage is more functional in nature.

Therefore independent testing alone can provide appropriate answers to the above questions. Following are the FIVE Commandments we identify from past experience for successfully executing independent package testing:




**1. Systematic and formal approach to test planning and design**

It goes without saying that any package tester should have an in-depth understanding of the out-of-the-box features of the package. Identification of how specific requirements are met (core vanilla, configuration or custom-coding within the product) is critical to ensure that the out-of-the box features are appropriately tested. Hence, a systematic and disciplined approach to planning the tests and designing the test conditions employed by testers' ensures optimal balancing of testing timelines and gives a truly vetted product.

**2. Align test plan as per business requirements and not as per the gaps identified**

When packages are implemented for a particular enterprise, requirements are analyzed for a good fit to the package and the gaps are captured as part of the gap analysis. The package is then configured or custom-coded to bridge the gaps. In situations where non-testers develop test plans, the test plans are generated according to the gaps.

This is usually the case when functional experts are conducting the testing. Planning the tests and designing the test cases against the requirements – and using the gaps as reference – ensures that testing aligns with the user's expectations as opposed to what IT had implemented.

**3. Keep the levels of testing small and simple**

Configuration includes setting up the business rules, access, workflow, and master data so as to ensure that the package is usable in day-to-day operations. Most effective methods for testing such changes would be all pairs / orthogonal array and boundary value analysis techniques. The custom-code development is fundamentally considered as traditionally developed. The right strategy is to keep it small and simple – small islands of functionalities are tested initially for a large set of scenarios before binding them to end-to-end workflows.

In other words, the testing strategy will progress from system-level testing (testing limited to a screen, report or module, etc.) to end-to-end business process testing.

a. The boundaries of system-level testing and end-to-end business process testing are usually fuzzy and easily confused.

b. Scope of test cases in system-level testing can be defined as three-point testing. It starts where data initialization is needed for a single user of system to do his/her work, moves to where the single user conducts the activity in the system, and ends where the single user is able to verify the results of his/her action.

c. Scope of test cases in end-to-end business process testing begins when data flows between multiple users and the functionality involves collaboration between these multiple users.

External Document © 2017 Infosys Limited

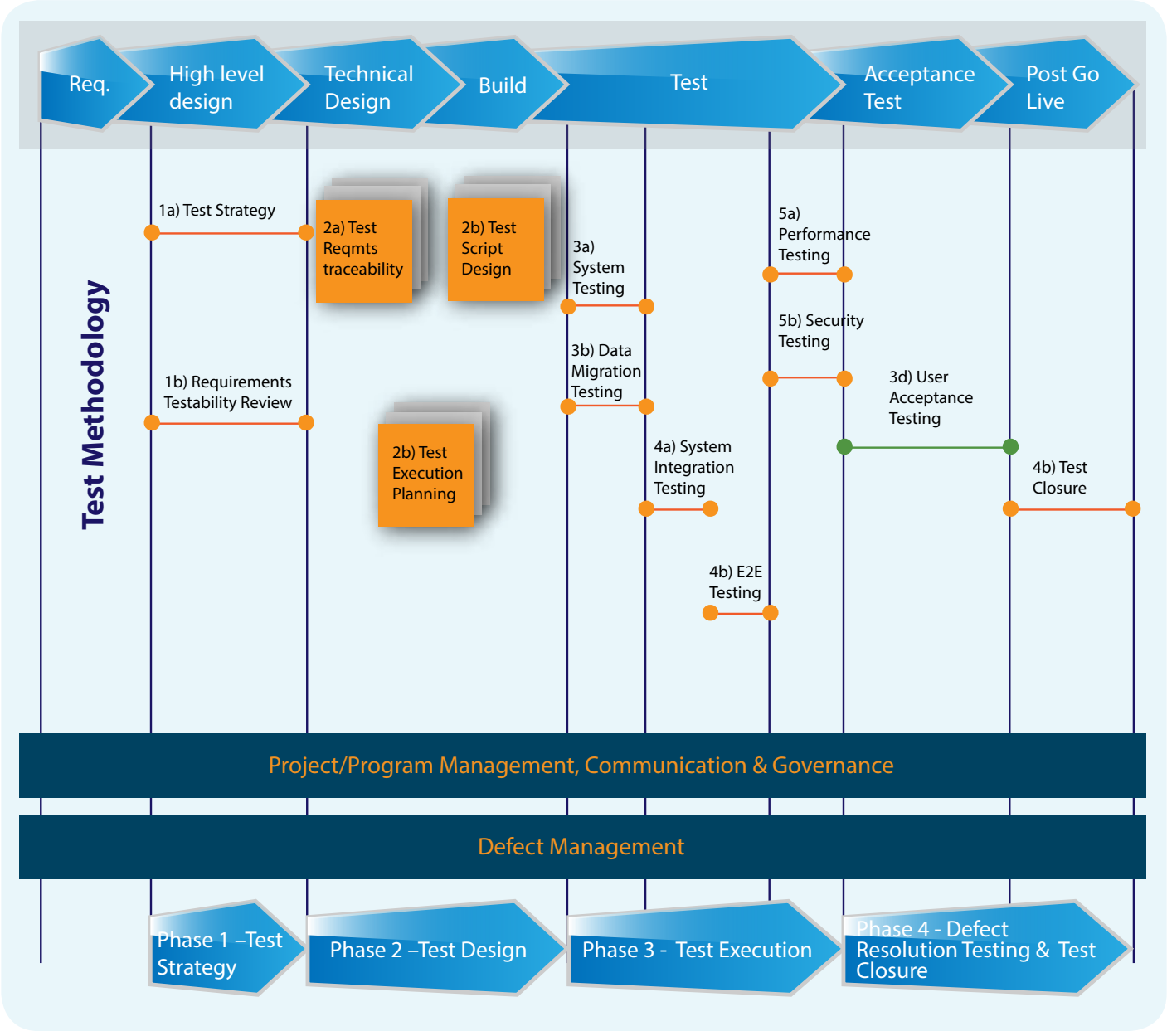
**4. Define and adopt structured test methodology**

The data flow from and to the package is arguably one of the most complex and risky dimension of package implementation/maintenance, which is compounded with the onset of new and advanced integration technologies (SOA, Cloud, etc.). In many cases, the packages get integrated with data warehouse (DW) and business intelligence (BI) systems.

These scenarios require a separate track comprising of testers who specialize in handling them – integration testers and DW/BI testers. The integration testers predominantly employ grey box testing methods as their testing strategy. The DW and BI testing track is best assigned to a specialist testing team, which deploys white box/ grey box testing techniques. It is recommended to compress the integration testing track between system testing and end-to-end business process testing (discussed in point 3 above).

The DW/BI testing is typically executed as an independent track with separate timelines altogether.

A pictorial representation of the test methodology adopted in relation to the implementation lifecycle is given for ease of understanding:




External Document © 2017 Infosys Limited

**5. Give special attention to activities that support and complement functional testing**

Another rationale of independent testing for package implementation/maintenance is the final principle of package testing. Besides the fact that functional testing for packages needs to be ably supported (Test Data Management, Environment Management and UAT Support), it also needs to be strengthened (Data Migration testing, Compliance testing etc.) and complemented (Non-functional Testing).

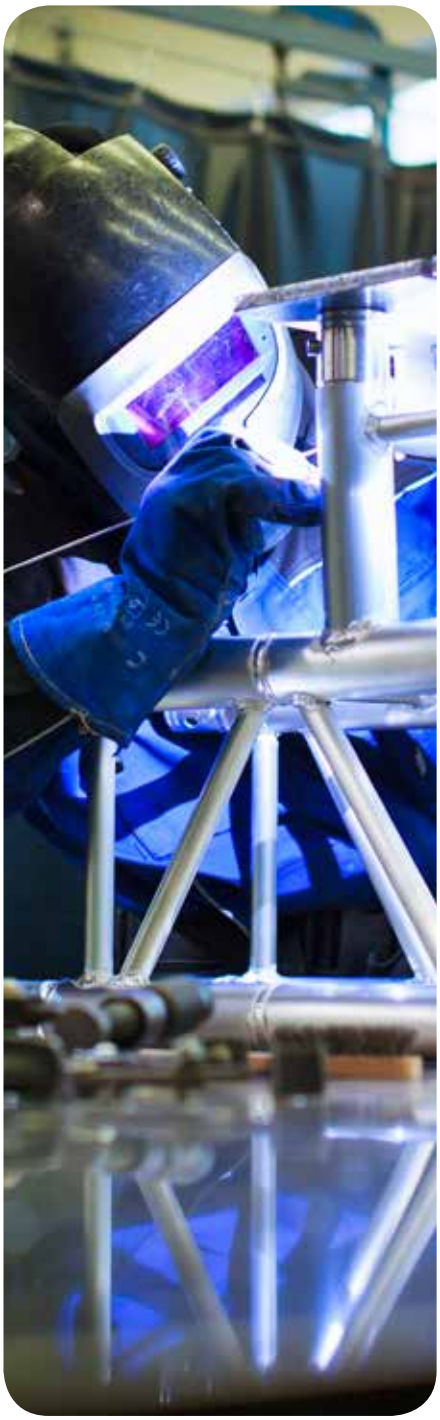
a. Test Data Management is one of the latest areas of specialization in the software testing function. With the database size handled by packages crossing from terabytes to petabytes, it is increasingly becoming an expensive and complex affair to create and/or maintain the master data and reference data required for a comprehensive testing exercise. A quick look at the number of non-defects reported by testing teams and users with root cause of incorrect test data will validate this claim. Companies are now setting up independent and centralized test data management teams, entrusted with the role of creating a repository of relevant business test data and provisioning to testing teams.

b. It is imperative that an adequate test environment is available for each of these testing activities. Creating multiple test environments is expensive. Sharing and reusing available infrastructure reduces the overall testing cost. However, multiple environments may be needed due to different environment requirements, data constraints, security constraints and time constraints. By sanctioning test environment budget to the independent testing team, companies make the testing teams accountable for this trade off. The independent testing team also allows for automatic control of the test environment, thereby streamlining efficient testing. On their part, the testing team will need to accept this responsibility by exercising due diligence in identifying test environments, providing concise environment requirements, and controlling the code deployments and data setup activities.

c. In cases where an existing legacy system is being retired and its functionality is being replaced with the package, data migration is required. Mostly there are several 'functional' issues inhibiting users to seamlessly switch over from legacy to the package, post implementation. Unless a specialized testing team focuses on data conversions and migration testing, these issues usually end up affecting user confidence and result in financial losses.

d. It is imperative that the package continues to comply with governmental and legal requirements like Sarbanes-Oxley (SOX). While traditionally these have been left for the users to test during UAT, the capability to automate several functionalities and the ability for cost-effective off-shoring is providing a justifiable argument to save the user's time and cost.

e. Last but not the least, performance testing for a package is different for custom or online applications. The data requirements, the need for functional knowledge during performance testing and the resource intensive middleware/backend programming necessitates a special and focused performance testing track.



External Document © 2017 Infosys Limited


## Summary

COTS/package testing is complicated and a difficult test effort to manage well. The test strategy for such implementations/ maintenance encompasses almost all of the specialized wings of testing function of today. It is important that companies recognize this and implement specialized and independent testing personnel/teams in supporting this activity for both the package implementation as well as on-going maintenance.


## About the Author


### Kiruba Vijayaraghavan

has more than 12 years of experience in testing across industry verticals and technologies. He specializes in the assessment and implementation of Test Centers of Excellence (TCoE), Test Factory for testing organizations and QA in Program Management for large scale implementations.


External Document © 2017 Infosys Limited

For more information, contact askus@infosys.com

© 2017 Infosys Limited, Bengaluru, India. All Rights Reserved. Infosys believes the information in this document is accurate as of its publication date; such information is subject to change without notice. Infosys acknowledges the proprietary rights of other companies to the trademarks, product names, and such other intellectual property rights mentioned in this document. Except as expressly permitted, neither this documentation nor any part of it may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, printing, photocopying, recording, or otherwise, without the prior permission of Infosys Limited and/or any named intellectual property rights holders under this document.