Organizing testing in relation to the user

Fifty years later

Fifty years ago, Han Urbanus was one of the founders of a new journal in the field of managing, controlling, organizing, and improving information assurance: *Compact*. In 1974, he wrote an article that elaborated on the role of the end user in testing a new (or modified) information system and how testing could be organized. Now, half a century later, we look back on the article: has its content stood the test of time?

Dennis Stam MSc CISA
EMITA
is a director within KPMG Advisory's Technology group.

For the Dutch version of this article, please scan the QR code.

INTRODUCTION

In 1974, Han Urbanus wrote an article for *Compact*, a new journal in the field of managing, controlling, organizing and improving information provision of which he was one of the founders. In the article ([Urba74]) Urbanus elaborated on the role of the end user in testing a new (or changed) information system and how testing could be organized.

This article first briefly summarizes Urbanus' insights on organizing user testing back in 1974. It then considers how testing is done today.

TESTING IN 1974

As early as 1974, there was a revolution in the world of software development. More and more reference was – rightly so – made to the responsibility that businesses have regarding the information in and from information systems. End users are therefore also crucial stakeholders who must give their approval for new systems (developed on their behalf) or changes to existing systems. The question, however, was: in what way can end users determine that the system has been developed in accordance with the requirements they have formulated?

Urbanus suggested that information systems testing should be considered a separate phase within system development. (Even) at that time, it was not unusual for testing a large and complex system to take at least as much time as designing and coding it.

In addition to considering the testing process as a separate phase, Urbanus also indicated that users should be involved throughout the development cycle. System acceptance criteria should be established during preliminary research. These criteria form the basis for the system design as well as the testing and (data) conversion activities. However, users are usually not familiar with testing and do not possess the skills to put together a test set. Users should therefore be involved in the process from the very beginning to "train" how to compose tests so that they provide maximum value for system and acceptance testing.

Not only was there a distinction between system and acceptance testing, Urbanus proposed (sub)phasing for system testing as early as 1974:

- testing of individual modules;
- testing of individual programs;
- Testing program series (chains);
- testing by subproject;
- testing of the entire project.

By testing small components first and – if the results are positive – scaling up to a larger whole and testing that, sub-phasing ensures that complexity decreases. This enables more careful testing and reduces the search for a needle in the (IT) haystack. Unfortunately, Urbanus noted, module testing is becoming less common due to impatience (people want to be able to test the full program as soon as possible) and savings.

Urbanus came up with a set of guidelines and the call to move toward a more organized and disciplined approach to testing:

- Formulate measures of program acceptance.
- Establish a test plan.
- Develop test cases.
- Describe the test data and its special purpose.
- Standardize job control of testing operations.
- Work on a basis of a source library to allow repetition.
- Determine in advance the expected results of testing.
- Develop utilities, such as for output comparison.
- Test methodically, keeping a close eye on what has and has not been tested.
- Keep a test log.
- Have data for system testing produced by users.

THE DEVELOPMENT OF PROFESSIONAL TESTING

Urbanus seems to be a pioneer in the field of test management. In the testing community, Glenford Myers is known as the first who, in 1979 (five years after Urbanus' article), gave a definition of "testing" in his book *The Art of Software Testing*:

> "(...) the process of executing a program with the intent of finding errors".

Urbanus seems to be a pioneer in the field of test management

However, the rapid development of information systems in the 1980s showed that the growing complexity demanded more than just "finding errors". A more structured approach was sought.

The first book on the TMap testing approach – developed on Dutch soil – was published in 1995. *Testen volgens TMap* ("Testing according to TMap") focused on defining testing as a structured process, introducing the following definition:

> "Testing is a process of planning, preparation and measurement aimed at determining the characteristics of an information system and demonstrating the difference between the current and desired status."

TMap brought the (IT) world a market standard for testing and test management. A few years later, in 1998, the International Software Testing Qualifications Board (ISTQB) was founded, an international organization that provides standardized training with tests and certificates for software testers.

In a world where IT was becoming increasingly important – if not indispensable – to organizations, the evolution of the testing approach was necessary. New insights into testing and system development, including the emergence of iterative and incremental development methodologies and Agile in particular, led to new versions of TMap. These newer versions brought a trade-off between costs and benefits (including by applying risk-based testing) and aligned with the innovations in system development.

A RETROSPECT ON THE ORIGINAL ARTICLE

Many of the propositions posited by Urbanus have become part of today's (professional) testing approach as proposed by TMap or ISTQB. Testing has become a separate phase, a distinction is made between system and acceptance testing, test plans are made in advance, users are involved in testing, et cetera. In short, testing has become a profession following a structured approach. Yet there are nuances between the way it is done today and the way it was done when Urbanus wrote his article.

The distinction in test phasing has been adopted. Nowadays, a distinction is usually made between development testing (by the developer), system testing (by test specialists) and acceptance testing (by acceptors). Within these tests, the principle proposed by Urbanus is also followed: start with testing small modules and slowly build up to testing integrations, programs and eventually chains. However, where Urbanus still observed a decline in the

application of development testing, many developers today recognize the importance of good unit testing and strive to (automatically) perform these tests with coverage of at least 80 percent of the program code present. This helps to detect errors early and avoid high costs for remediation in later phases of system development.

Similar to the rise of unit testing, the development of "continuous integration" (CI) and "continuous delivery" (CD) has provided structure to how software code is managed (using a version management system) on the one hand and a repeatable process on the other. Changes are no longer made directly to the information system, instead, the source code is maintained in a version management system such as Git or SVN. A new version of the software can be prepared and compiled from this system and rolled out to a (test) environment. This gives the tester more clarity about (the version of) the test object, and newer versions – with, for example, the latest fixes – can be made available more easily for testing. In addition, this setup provides a basis for the automated execution of unit and (part of) system tests, which also offers more certainty about the quality of the test object when the tester starts working with a new version.

The role of the user organization is also formalized; in almost every test program, representatives of the business are asked to participate in testing in addition to the use of test professionals. In some cases, end users are even asked to participate in reviewing the design already during system development and performing tests during the system test phase. Today, however, when performing acceptance tests, a distinction is made between different acceptance testing types, such as functional acceptance test (FAT), user acceptance test (UAT), production acceptance test (PAT) and operational acceptance test (OAT). In this context, users are the implementers within the UAT (and administrators, also to be considered a special user group, within the OAT), with the understanding that users must – above all – bring in subject matter expertise. For this reason, users are often supported by test specialists in managing test preparation and execution, defining the test set and producing the test data.

The growing complexity of information systems has brought similar complexity to the data used. Whereas in 1974 there was still simple talk of users preparing test data, modern systems require a multitude of efforts to arrive at a good set of test data. After all, testers need test data focused on situations that have not (yet) occurred, for which test data must be fabricated, as well as test data that are as representative as possible of real-life situations that the information system has to deal with. The emergence of privacy legislation such as GDPR complicates the latter in recent years; data that exists in the production environment often contains privacy-sensitive data and was not collected for the purpose of testing, and should not be used for testing for that reason. Test data management is an emerging specialization focused on crafting data for non-production purposes and involves anonymizing, subsetting, masking, scrubbing, "profiling" and "ageing" transactions and "provisioning" data.

In addition to a growing need to manage test data, the frequency of delivery and the increasing complexity of information systems require an ever-increasing degree of test automation and deployment of tooling. The "utilities" mentioned by Urbanus are therefore no longer small utilities for output comparison or taking snapshots, but fully automated test sets, emulation of software components (service virtualization) and tools to run tests on multiple platforms simultaneously (for testing mobile apps or Web applications, for example).

CONCLUSION

All in all, the world of testing has not stood still in the past fifty years. Urbanus was rightfully a visionary in this field, as evidenced by the multitude of guidelines he drafted back in 1974 that eventually became part of the market standards for testing and test management in the 1990s.

That said, however, we are not there yet. As information systems become even more complex and new system development methods evolve, the testing domain also has to keep up. The emergence of new trends such as big data, blockchain and even program generation using AI such as ChatGPT will require further evolution of testing. Hopefully fifty years from now, a retrospect on this article will draw a similar conclusion, and testing will have been further professionalized.

Literature

[Urba74] Urbanus, J.H. (1974). De organisatie van het testen in relatie tot de gebruiker. *Compact, 1974*(1), 3-9.

About the author

Dennis Stam MSc CISA EMITA is a director at KPMG Advisory N.V. He is responsible for the Tech Advisory team, which focuses on improving software quality in a broad sense. He specializes mainly in the topics of source code review, testing and test management, and has extensive experience with other aspects within system development projects.