9

March, 2010

testing experience

The Magazine for Professional Testers

printed in Germany

print version 8,00 €

free digital version

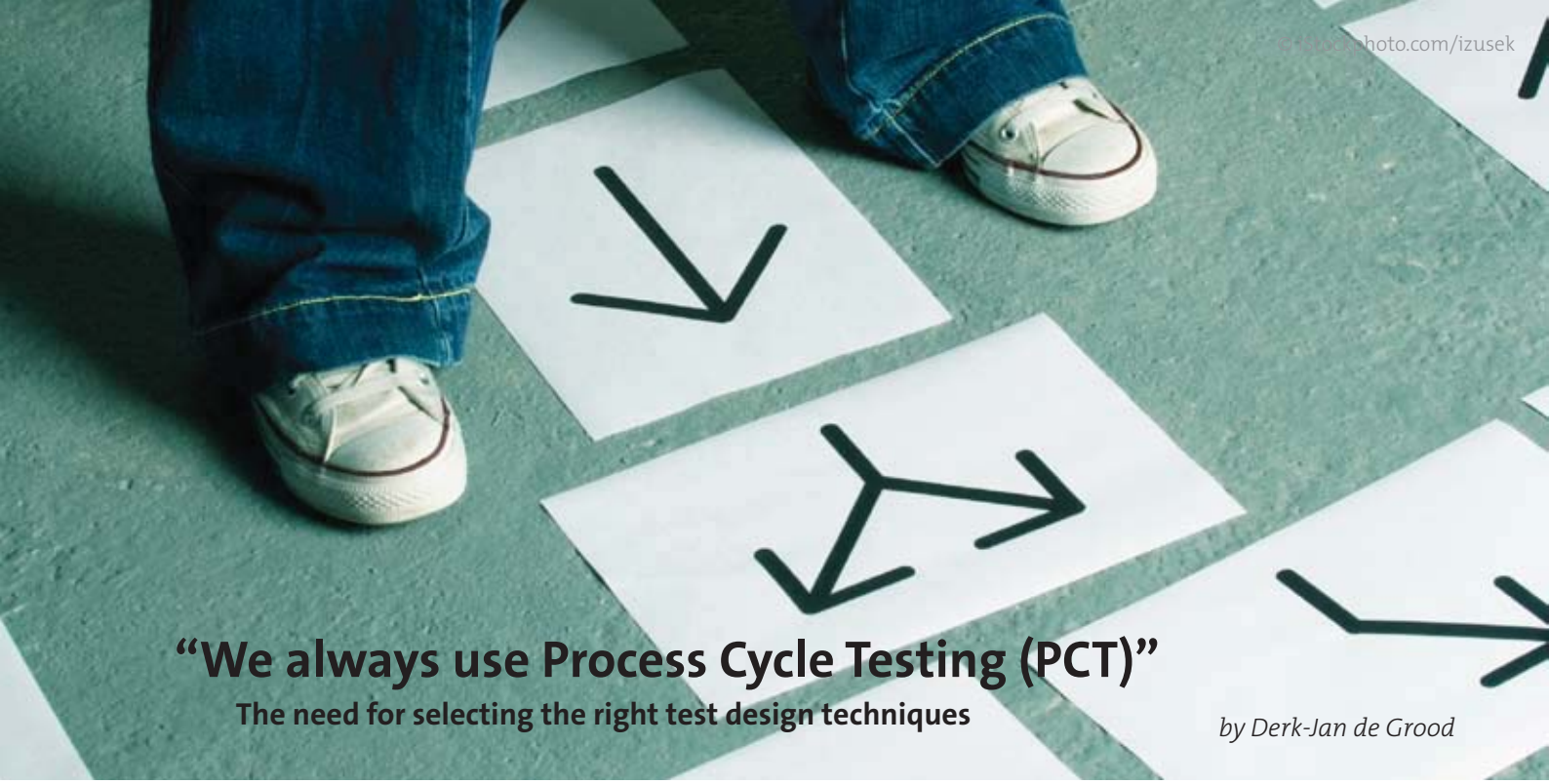
www.testingexperience.com

ISSN 1866-5705

Advanced Testing Techniques

iqnite Conferences Special

GERMANY 2010



"We always use Process Cycle Testing (PCT)"

The need for selecting the right test design techniques

by Derk-Jan de Grood

I once introduced test design techniques to my test department. We trained the testers and made "quick reference cards", explaining how each technique was to be used. Still, my testers were clearly struggling with the application of the techniques. When I sat down with one of the testers, it became clear why. In this specific case, my colleague derived test cases for the main scenario of a use case. He used a decision table. Each of the negative conditions in the decision table led him to the alternatives and exceptions that had already been defined in the remainder of the use case. I explained him that this way the technique was a useful review technique that could help to check the completeness of the use case. It was, however, not effective for defining test cases. To cut a long story short, he tried to apply the wrong technique on the wrong part of the specification at the wrong time.

We testers put a lot of emphasis on using test design techniques. Just to give an indication, TMap Next [TMap,2006] dedicates about 130 pages to test design techniques. This is about 17% of the whole book. "Foundations of software testing" [ISTQB, 2007] by Graham et al., dedicates about 20% of its content to explaining the functioning of different techniques. Syntax testing, Boundary Value Analysis (BVA) and State Transition Testing (STT) are typical examples. But there are many more techniques to aid the tester. Strangely enough, in most books, how to select a technique is hardly discussed. In my opinion, that is a shame, because selecting the right test technique is important for the efficiency of testing. My experience is that many testers find it difficult to select the right techniques.

Solid brick wall

Although we may have difficulties with the selection process, we generally agree that test techniques have great advantages. This is regardless of the techniques being used in a formal scripted way, or in the tester's mind while doing exploratory testing. Since each test design technique has its own focus, applying that technique will increase the probability that errors of a certain type will be found. The structural approach of the technique ensures that the work is done more thoroughly and that less obvious situations will be tested. It will therefore gain a more thorough understanding of the quality of the system and most likely result in errors found that would have remained unnoticed otherwise.

I recall a discussion that I once had with a test manager at a large insurance company. When addressing the problem of selecting the right techniques, she smiled and presented me with her solution; "We always use PCT". She had no problems with making a selection at all. This was a disturbing situation to me, since there is not one technique that is always the most effective. Unfortunately, the situation at that specific company is not unique. Many organizations describe a limited set of techniques in their test strategy, and these are used over and over again. I have heard of testers who tried to use different techniques for good reasons, and ran into solid brick walls.

Testing contributes to better product quality and trust in the system. As testers we should be well aware of the goals that the organization has with the product. What are the business drivers to which it contributes? What problems does it help solving? The testing process starts with learning about the goals of our stakeholders. Risk analysis is often used as a method to identify the threats to the anticipated goals and to understand what types of errors the business does not want to encounter during daily operations. Risk analysis identifies what information is relevant to the stakeholders. Test techniques can be applied in order to translate the risks, threats and unwanted errors, that have been determined, into well argumented test actions. By using the right techniques, the tester executes the right tests and is accountable for his actions and the quality of his advice. Since the techniques define the tests that are executed, they also determine the information that can be given to the stakeholders.

Stakeholder attention during triage meetings

Selecting the techniques to be used is a careful process. Selecting the wrong technique may cost you a lot of time without finding many errors or useful information. Failing to select the right technique may lead to major defects not being found (see textbox). It is important to link the techniques with the risk they cover and the type of errors they help to find. In order to increase this awareness, I often show bug reports in my training sessions. The attendees are asked to name test design techniques that could have been used to find the error. To understand this has a great advantage. If you can predict from experience that a certain bug report will not get a high priority and is likely to be put aside, do not use the techniques that will help you find these types of errors. You may know what kind of bug reports will cause commotion in the organization and get stakeholder attention during triage meetings. Select design techniques that will help you find those errors.

Test design technique effectiveness

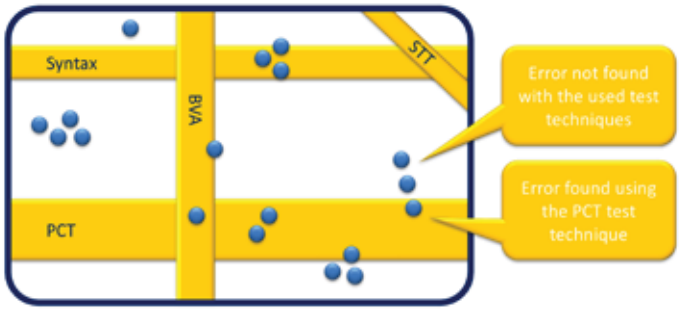


Figure 1
In the above figure, the test object is represented by the outer frame. The test object contains a number of errors, but we do not know where they are. If we did, we would not have to test, would we? The bands indicate the part of the code that is covered with test cases that were derived using test techniques. In this example four techniques are used, Syntax testing, Boundary Value Analysis (BVA), Process Cycle Testing (PCT) and State Transition Testing (STT). While executing the defined test cases we stumble upon several errors. From the example we can learn that

- Errors that are found by one technique are not necessarily found by another technique.
- Some techniques are very suitable in a specific situation and result in more errors found than others. In this example, using PCT has resulted in five errors, compared to STT none.
- Some errors cannot be found with any of the chosen techniques.

If you don't know what types of errors will catch your stakeholders attention, it's time to pay them a visit. Like we said before, typical risk analysis will lead to understanding this. The PRIMA method [SmarTEST, 2008], as an example, links the priority of the functions with 'types of errors'. This gives you precisely the information that you need; an indication of errors to look for and a measure of how important it is to find these.

With the PRIMA matrix, the first half of the puzzle is solved. It helps to answer the two questions: "What kind of errors do you want me to find?" and "What impact do these errors have in the production environment? ". The questions are included in the decision model that is depicted in figure 3.

Not because you can, but ...

Many times test techniques are applied because the tester is familiar with them and the test basis supports the technique. Using the decision model, we inverse the problem. We do not use techniques because we can, but because they help us to provide the right information. Unlike many methods we start with the errors and select the techniques that help to find them. In my book [TestGoal, 2008] I present a table that enables the tester to select the right techniques. The table below shows a few examples.

One other question needs to be answered. Is the required information available? We can choose a technique, but the required information needs to be available. You can decide to do state tran-

PRIMA product Risico Matrix — **Project: CRM implementatie**

Functions in the system

Types of Errors (Domains)

Kwaliteit top-10	100	24	7	9	17	12	13	5	12
1. Functionaliteit	36	xxx	x		xx	x			x
2. Proces aansluiting	16	xxx	xxx	xx	x	xx	xxx		xxx
3. Bedienbaarheid	6	xxx	x		x	xx			
4. Tijdsaspecten	6	xx		xx	xx	x	xxx		
5. Foutbestendigheid	6	xx	x	xx	x		xx		
6. Efficiëntie	4	x			xx	xx			xxx
7. Beschikbaarheid	4	xx		x	xxx				xx
8. Traceerbaarheid	4	x	xx		xxx	x	xx	xxx	xx
9. Beheerbaarheid	4		x		xx		xx		x
10. Integriteit	4			x	xx	xx	xxx	xxx	

Figure 2

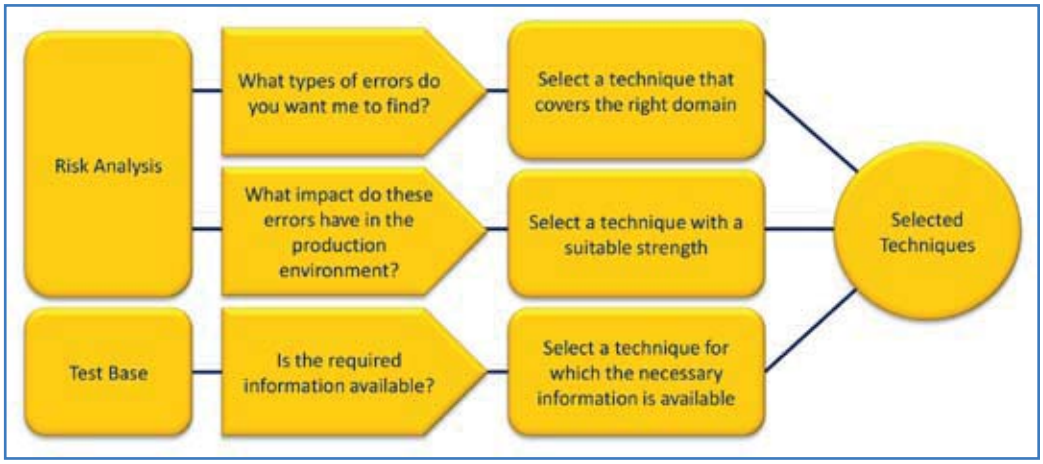


Figure 3

Errors	Technique	Required information
• Processing errors due to invalid inputs, e.g. invalid values, wrong units, etc. • Failing DB inserts due to invalid validation at GUI-message interface	• Syntax Testing	• Data dictionary explaining restrictions for each data element including conditional validation rules.
• Wrong functional decisions in for example business rules or recursive processing	• Equivalence Partitioning (EP) • Boundary Value Analysis (BVA) • Decision table testing (C/E)	• Defined decision points and conditions under which tasks are executed.
• Process breaks down and cannot be completed correctly • Process steps are taken in wrong order • Entities in the system can get into wrong state • System notifications are sent at the wrong moment or not at all • Unauthorized user is able to fulfill part of a process.	• Process Cycle Testing (PCT) • State Transition Testing (STT)	• Process description or swim lane diagrams • STD • Conditions under which certain transitions or process steps are taken • Authorization matrix

sition testing, but will you draw the necessary STDs if these are not provided? The table indicates a few examples of information that are required to apply the technique. There are two options when the test basis does not contain enough information:

- Find the information, no matter how hard it is
- Do not use the selected test design technique

The first option is preferable. Missing information or information that is hard to find is a risk. If you cannot find the information, other people involved in the project will probably have the same problem. Who knows which system behavior will be implemented, who will decide whether the application is good enough? Ensuring that the information becomes available reduces the risk of surprises arising later on in the software development project. But it's not always possible to obtain the missing information. In this case, the function will have to be tested in a different way, using less suitable or no techniques at all. It is important that this is communicated clearly; after all, some risks will not be addressed as intended and the information provided by the test project may be less precise and not take away all uncertainties.

Conclusion

Although there is much information available on test design techniques, very little is being said on how to select the right design techniques. The errors that you do not want to have in your production system should be the starting point when making your selection. Testers should therefore understand the relation between types of errors and techniques that help finding them. This understanding will lead to a better use of techniques and will lead to asking the right questions to the stakeholders.

Technique selection deserves more attention. I made a start by including it in my book. Who will take the challenge ? It would be a success if the selection of test techniques is explicitly addressed in the next test process that you'll implement, or in the next book that you'll publish. Any test manager or team manager that is ambitious to integrate the selection as part of the team's standard process is invited to mail me with his success or challenges.

References

- [TestGoal, 2008] TestGoal, result driven testing, Derk-Jan de Grood, Springer 2008, ISBN 9783540788287
- [SmarTEST, 2008] SmarTEST, Egbert Bouman, Academic services 2008, ISBN 9789012125970
- [TMap, 2006] TMap Next, Tim Koomen et. al., Tutein Nolthenius 2006 ISBN 9789072194794
- [ISTQB, 2007] Foundations of Software Testing, Dorothy Graham et.al., Thomson, 2007, ISBN 9781844803552

With many thanks to Egbert Bouman, Guido Dulos, Martin Kooij and Esther Martin.

Biography

Derk-Jan de Grood has broad, hands-on experience as test engineer, test manager and advisor in a large range of industries. As manager of various test departments he has learnt how to implement test methods the practical way. He gives lectures at various Dutch universities and is author of the first educational book specially written for teaching software testing at Dutch universities. Derk-Jan is also author of "TestGoal, the result-driven test philosophy".
As ISTQB advanced certified test manager and an ISEB accredited trainer, he provides training sessions on a regular basis. These training sessions vary from the standard introduction into result-driven testing to custom-made training courses that tune in on specific needs of the client. Besides that, he is a passionate, inspiring speaker at major testing conferences all over the world, such as the STAR conferences in the USA and Europe.

Recently, Derk-Jan has joined Valori. With that he started a new period in his career. He is in the program board of the 1st Dutch Test Conference. His new book on 'how to give a good presentation' has just been released and he started a column on testnieuws.nl.