

TestCase Design: a practical guide to building the most powerful test cases

Tutorium – TestNet Summer School 2011

July 13th, 2011 | Wolfgang Platz

Agenda | *proposal*

- **Block I – 9:00 – 10:45 ... Problem definition & methodological basis**
 - An intro to the participants and *TRICENTIS*®
 - Starting situation – presentation of the problem
 - Methodological basis of the TestCase and test data design
- <Break>
- **Block II – 11:00 – 13:00 ... TestCase and test data design**
 - Example: fictitious task from the banking sector
 - Summary



Easy Testmanagement & -automation

Methodology/Conceptual

Test Planning
Test Case Finding/Design
Test Case Automation
Test Execution/-management

Founded in: 1997
Headquarters: Vienna
Mitarbeiter: 130+
Customers: 220+
Operating in: Austria
Germany
Switzerland
Netherlands
UK
Australia/NZ
USA



Services

Consulting
Trainings
Project Support
Test as a Managed Service

Solutions

TOSCA Commander™: Test Management
TOSCA Adapter: Test Automation
TOSCA TestCase Design: Test Data Generation

Education

- Graduated with honors in Technical Physics at the University of Technology in Vienna
- Studies in Business management, Law and Economics
- Certified management consultant

Career

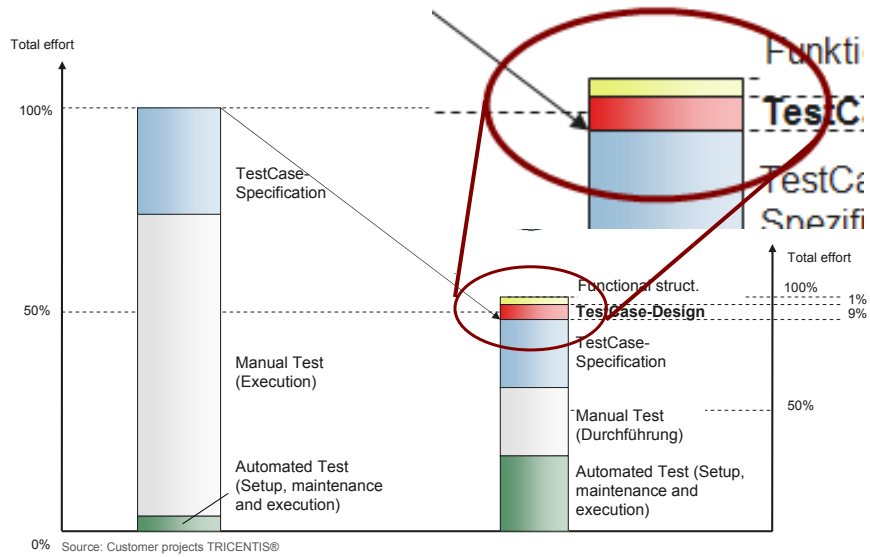
- 1997 TRICENTIS Technology & Consulting GmbH (Founder, CEO)
- 1995 - 1997 Cap Gemini Ernst & Young (Consultant)
- 1992 - 1995 VKI Verein für Konsumenten-Information (Developer)



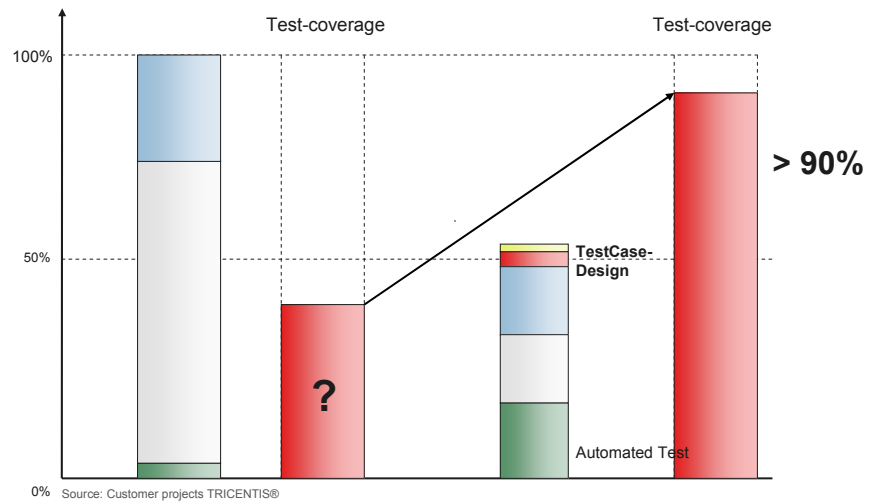
Relevant project experience

- Conception, coordination and support with implementing the dynamic synthetic test data concept *TOSCA @data* in the banking sector
- Test concept, TestCase-Design, test management at about 30 different customers in the following sectors: insurance, banking, telcos, stock exchange, commerce, energy, industry, etc. (conception, moderation and management) of international range and importance
- Technical & business-based concept, architecture and implementation of the TOSCA Explorer (predecessor product of the newly developed *TOSCA Testsuite™*)
- Technical & business-based concept, architecture and implementation of the product definition machine of one of the world's biggest insurance companies (project manager, Chief Architect)

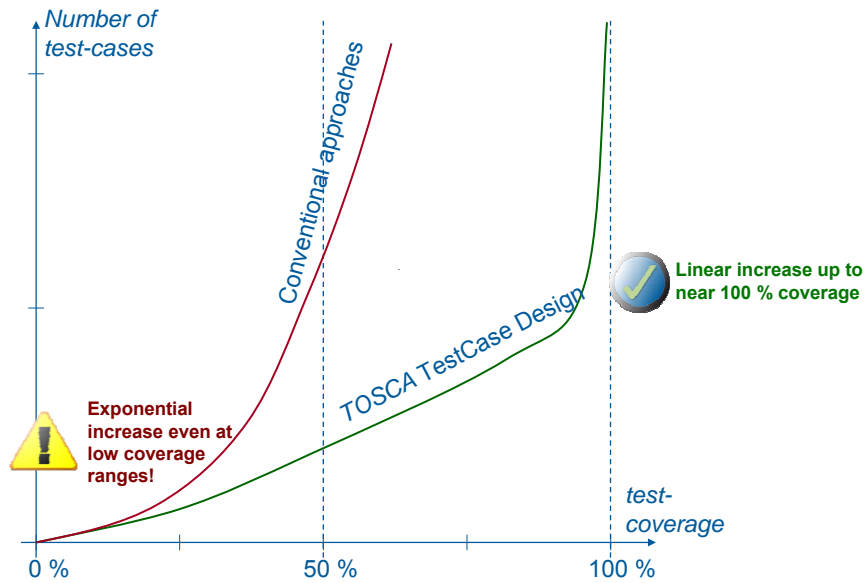
Efforts in test projects



Increased test-coverage



Number of test cases and coverage



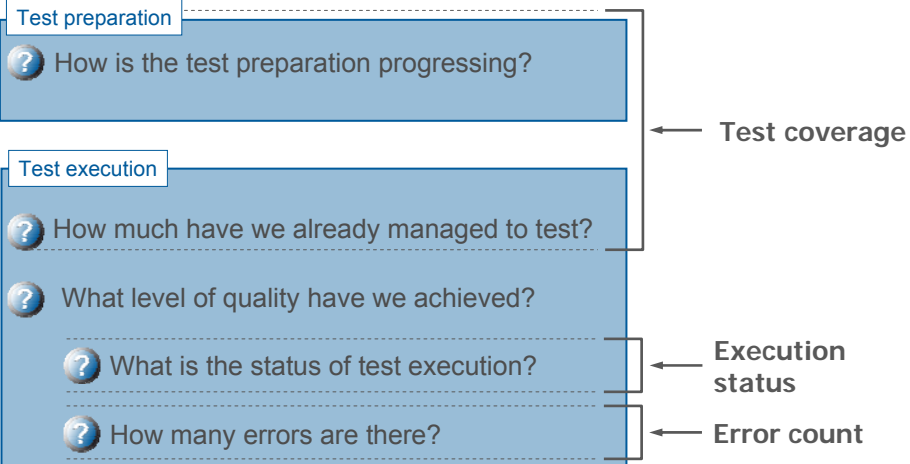
Requirements for the test

"We expect a reliable statement on whether our new software (version) can go into production."

What are the requirements for this statement?

- Simple and comprehensive
- Traceable and verifiable at all times
- Reliable

Which basic questions should the test process answer?



Test coverage [%] states the extent to which the entire functionality under test can be or was tested.

Absolute test coverage

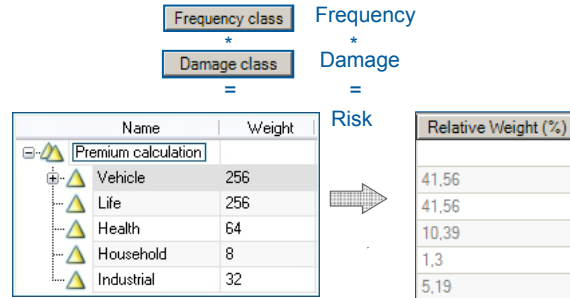
- The quotient of the functions tested and all the testing functions
- All the functions under test are assessed in the same way.

⚠ Relative test coverage

- Risk assessment of all the functions under test
- Test coverage = sum of risk weights of the functions tested

1 Task 1: We must know the value of the **functions under test!**

Risk weighting level by level



Structure for calculating insurance premiums

Weights are standardized to 1.

- Risk weight = frequency x expected damage
- Weighting is done in levels from the top down to the basic function.

Risk weighting level by level (cont.)

Name	Weight	Relative Weight (%)	Contribution (%)
Premium calculation		41,56	41,56
Vehicle	256	41,56	41,56
Liability	512	77,73	30,22
Comprehensive	128	19,18	7,56
Passenger	64	9,09	3,78
Life	256	41,56	41,56
Health	64	10,39	10,39
Household	8	1,3	1,3
Industrial	32	5,19	5,19

- The contribution to the entire risk is determined for each function.

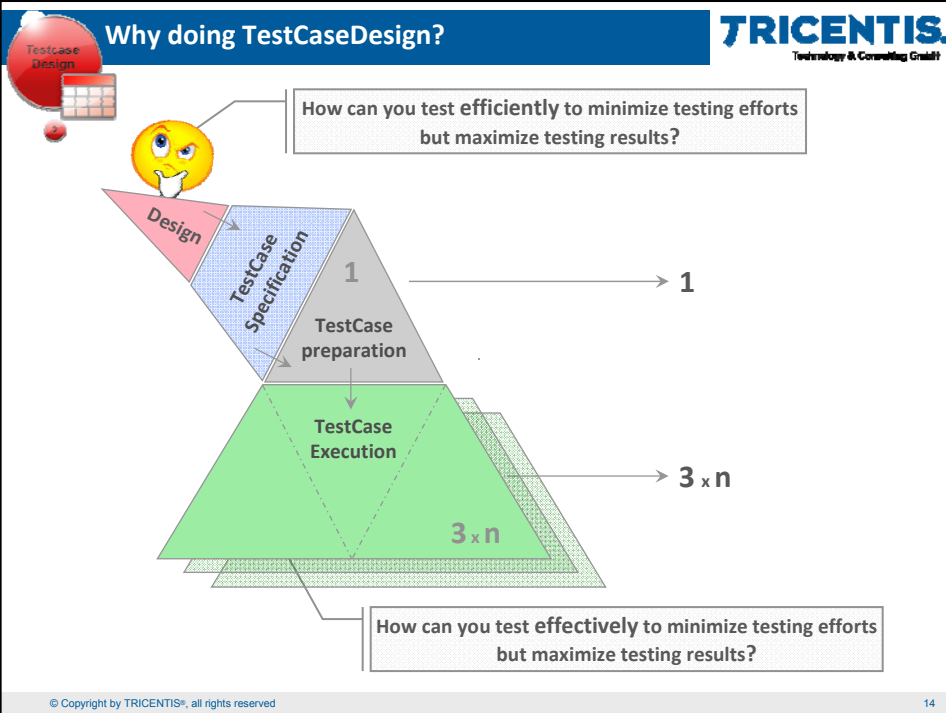
2 Task 2: We need test cases for the functions.

Measurable statements | Conclusion


Name	Execution State (%)		
BI	36	8	48
└─ Extraktor	50	6	32
└─ LDD (Local Data Delivery)	18	20	62
└─ GDP (Group Datapool)	33	17	50
└─ File Input Layer -> Staging Layer 1 (SA_)			100
└─ SA_ -> Staging Layer 2 (S2_)			100
└─ S2_bzw. LDD_ -> Business Layer (B2_)	37	19	44
└─ Kalkulation	42	21	37
└─ EUR Umrechnungen	67		33
└─ Ableitung Rating-Datum			100
└─ Fund Splitting	67		33
└─ Konsolidierung Sicherheiten			100
└─ Transformation			100
└─ Datenveredelung			100
└─ Export & Reporting			100
└─ SAS Exposure Calc	60		40
└─ SAS Collateral Allocation	33	17	50
└─ SAS RWA Engine	34	17	49
└─ Meldewesen (ANATOL, ROM/COREP, D...)	51		49
└─ Risk Reporting	24	18	33
└─ GPM/OFSA	32		68
└─ GPM Reporting	44	22	34

Example of functional structure for a DWH/BI-application at Austrian Bank, extract from *TOSCA Testsuite™*

Why doing TestCaseDesign?



Why doing TestCaseDesign?



How can you test efficiently to minimize testing efforts but maximize testing results?


The diagram illustrates a pyramid of testing stages. The top layer is 'Design' (1), followed by 'Test Case Specification' (1), 'Test Case preparation' (1), and 'Test Case Execution' (3 x n). A yellow warning icon points to the top layer, indicating the goal of the design phase.

- ! lowest possible number of test cases
- highest possible test coverage
- little effort in error analysis
- minimize redundant test case errors

© Copyright by TRICENTIS®, all rights reserved

15

Example | Annual rate of a vehicle insurance




The chart shows the annual rate of a vehicle insurance based on age groups. The rates are: <18 (N.A.), 18..23 (+20%), >59 (-10%), and another group (-5%). Silhouettes of people represent each age group: children for <18, a couple for 18..23, an elderly couple for >59, and a woman for the final group.

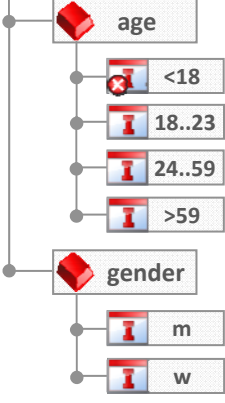
© Copyright by TRICENTIS®, all rights reserved

16

How do we combine age and gender?



Technology & Consulting GmbH



The decision tree starts with 'age' and branches into four categories: '<18' (marked with a red X), '18..23', '24..59', and '>59'. The 'age' node is marked with a red diamond. The 'gender' node is also marked with a red diamond and branches into 'm' and 'w'.

Select Case age

Case Is < 18 ' no insurance

 break;

Case 18 To 23 ' 20% additional charge

 factor_age=1.2

Case 24 To 59 ' normal

 factor_age=1.0

Case Is > 59 ' 10% discount

 factor_age=0.9

End Select

Select Case gender

Case "male" ' default rate

 factor_gender=1.0

Case "female" ' 5% women's discount


 factor_gender=0.95

End Select

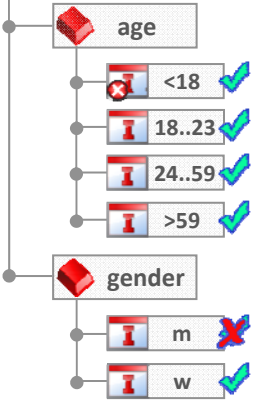
© Copyright by TRICENTIS®, all rights reserved

17

How do we combine age and gender?



Technology & Consulting GmbH

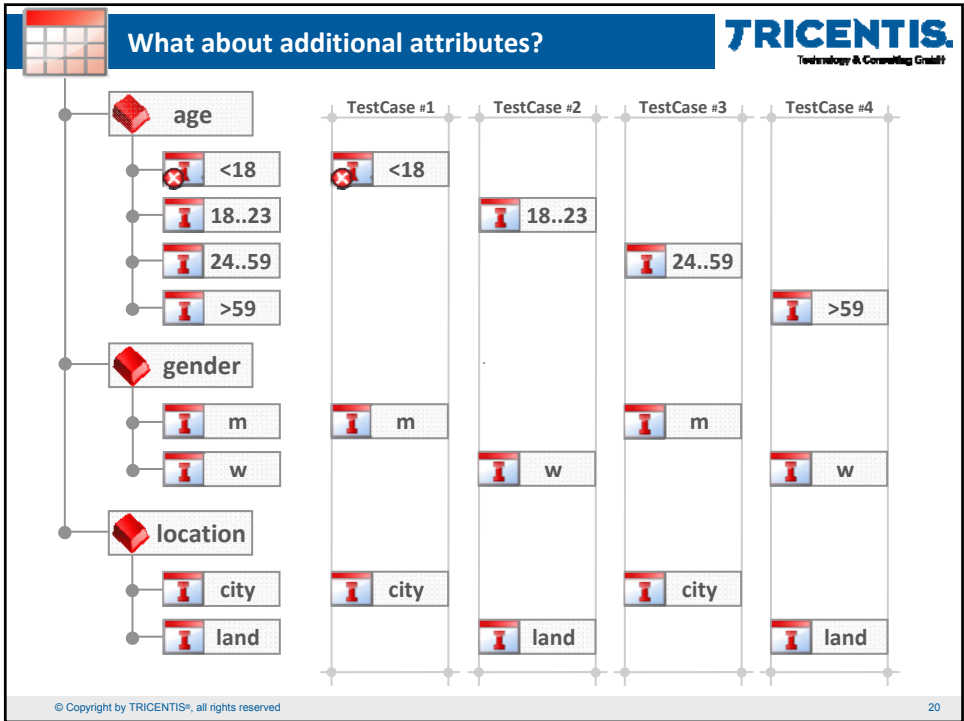
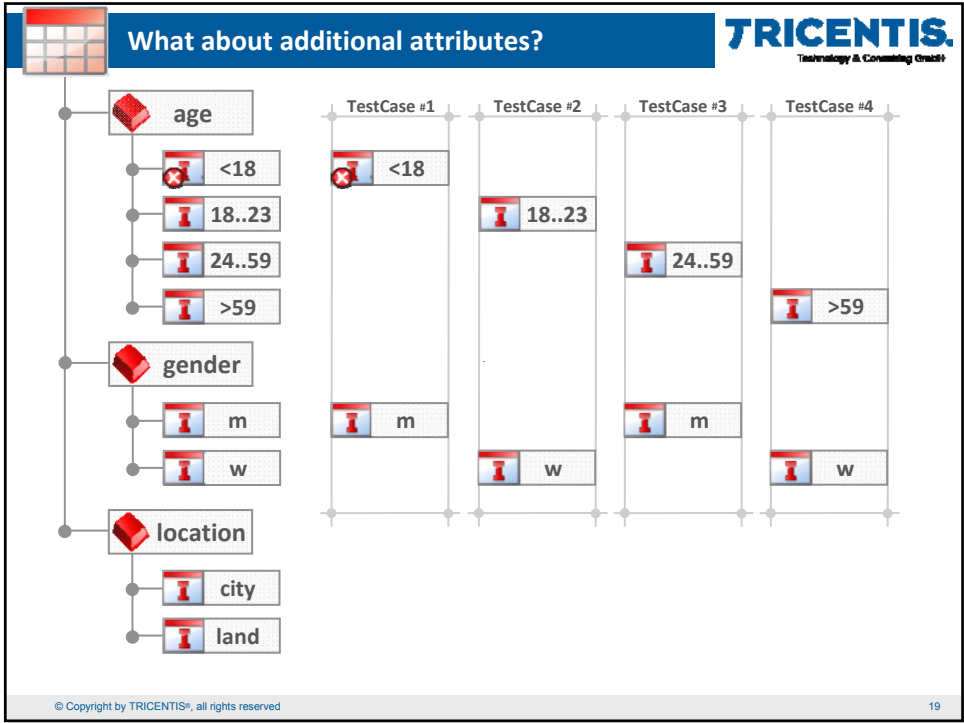


The decision tree is identical to slide 17, but with green checkmarks next to the '<18', '18..23', '24..59', and '>59' age nodes, and a red X next to the 'm' gender node.

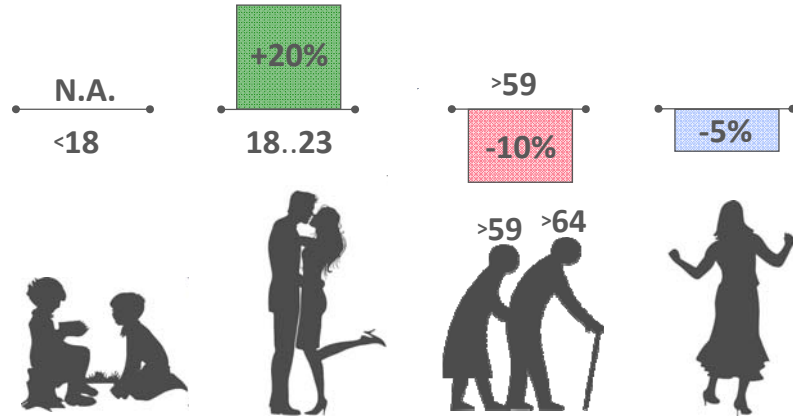
	TestCase #1	TestCase #2	TestCase #3	TestCase #4
age <18				
age 18..23				
age 24..59				
age >59				
gender m				
gender w				

© Copyright by TRICENTIS®, all rights reserved

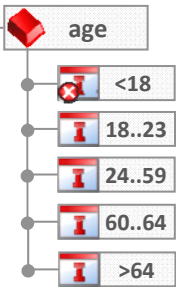
18



Example | Extension of the topic



Extension of the topic!



```
Select Case age
Case Is < 18 ' no driver's license
' ...
Case 18 To 23 ' young driver's license
factor_age = 1.2
Case 24 To 59 ' normal
factor_age = 1.0
Case 60 To 64 ' senior discount; male <> female
Select Case gender
Case "male"
factor_age = 1.0
Case "female"
factor_age = 0.9
End Select
Case Is > 64 ' senior discount; all
factor_age = 0.9
End Select
```

TestCase #1	TestCase #2	TestCase #3	TestCase #4	TestCase #5	TestCase #6
<18	18..23	24..59	60..64	60..64	>64
m	w	m	w	m	w
city	land	city	land	city	land

Boundaries **TRICENTIS**
Technology & Consulting GmbH

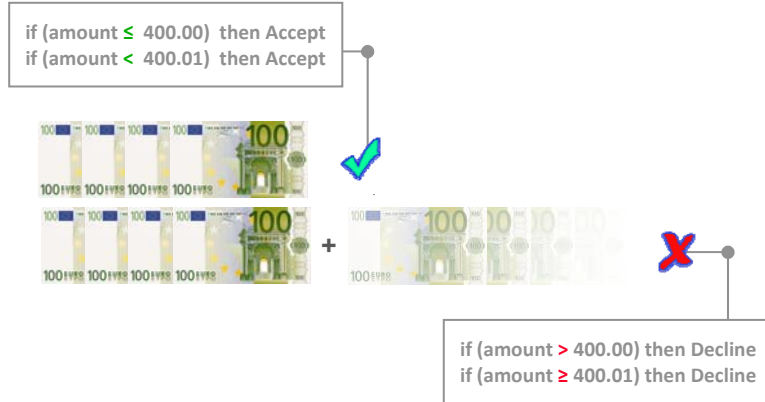
The diagram illustrates boundary values for two attributes: 'age' and 'gender'. The 'age' attribute is shown with four ranges: <18, 18..23, 24..59, and >59. The 'gender' attribute is shown with two values: 'm' and 'w'. A box with a thinking emoji asks 'Boundary values?' with a dashed arrow pointing to the <18 range.

© Copyright by TRICENTIS®, all rights reserved 23

Equivalence partitioning **TRICENTIS**
Technology & Consulting GmbH




The diagram shows a number line with points $-\infty$, 1, 99, and ∞ . Three intervals are defined: <1 , $1..99$, and >99 . A box with a thinking emoji asks 'Should we care about the boundary values?' with a line pointing to the <1 interval.

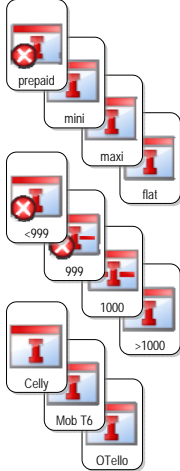
© Copyright by TRICENTIS®, all rights reserved 24



- Orthogonality is of highest importance to the attributes (especially with business applications)
 - Most attributes are orthogonal.
 - If orthogonality does not apply, it usually only refers to some equivalence classes of attributes, but not to all of its instances.
- However if applied consequently to too many attributes, it shows substantial weaknesses:
 - The verification of results is not provided - **different combinations could possibly show identical results.**
 - There is a higher probability of errors with increasing complexity - this decreases the probability that TestCases are executed completely.
 - **Error analysis increases in complexity.**

Cardgame

- basic monthly fee 
- bonus points 
- mobile phone 

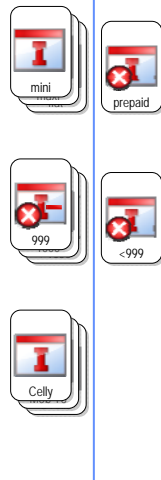
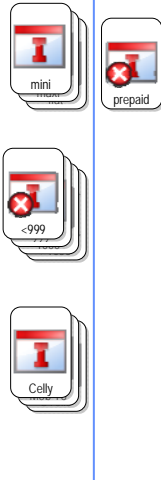


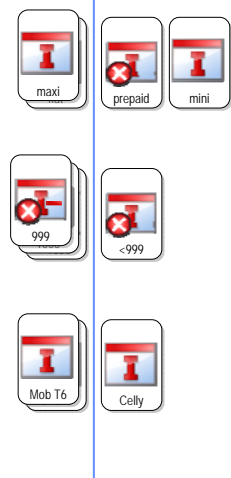
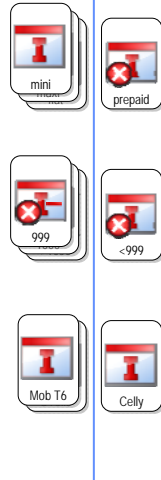
The cards are arranged in a vertical stack. The top card is 'prepaid' with a red 'X' icon. Below it are 'mini', 'maxi', and 'flat' with red 'I' icons. The next card is '<999' with a red 'X' icon. Below that are '999', '1000', and '>1000' with red 'I' icons. The bottom cards are 'Celly', 'Mob T6', and 'OTello' with red 'I' icons.

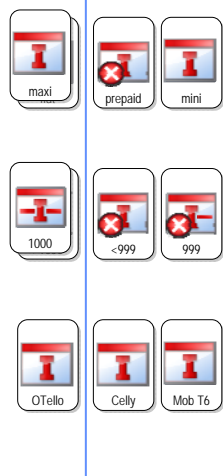
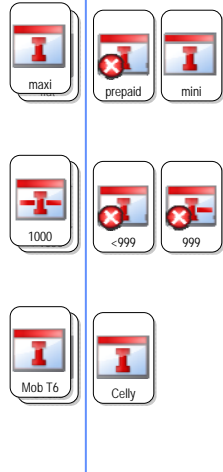
Ortho

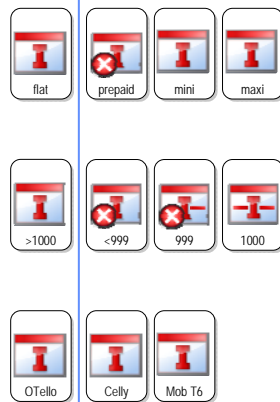
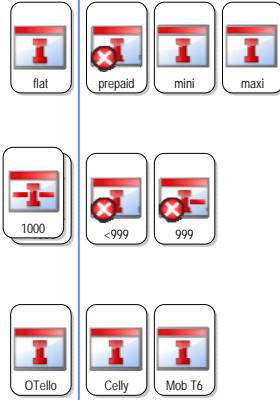


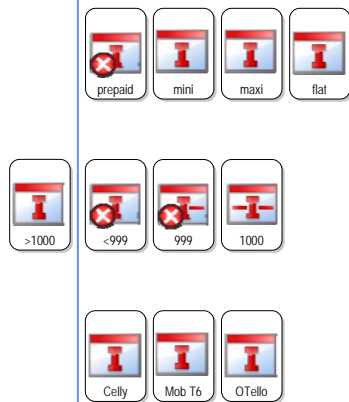
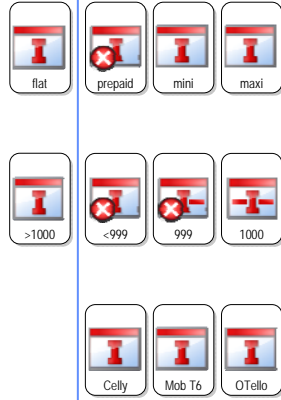
Three stacks of cards are shown on the left side of the slide, separated by a vertical blue line. The top stack is labeled 'prepaid' with a red 'X' icon. The middle stack is labeled '<999' with a red 'X' icon. The bottom stack is labeled 'Celly' with a red 'I' icon.

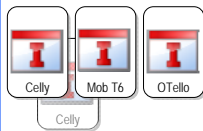


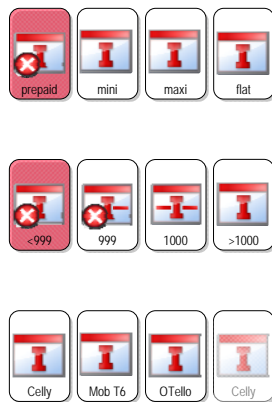
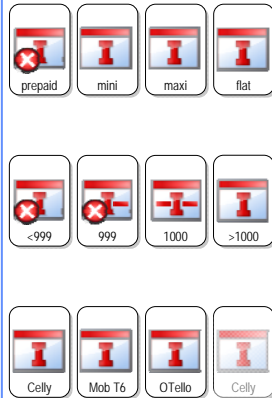












- + number of testcases
 - principle of cardinality= 4
 - all combinations:
 $4 \cdot 4 \cdot 3 = 48$
- ambiguous hotspot (focus)
- invalid instances?
- boundary values?

Straight Through criteria...

1

Risk

Maximum risk means maximum coverage!

2

Ease

Minimal functionality means quick implementation!

3

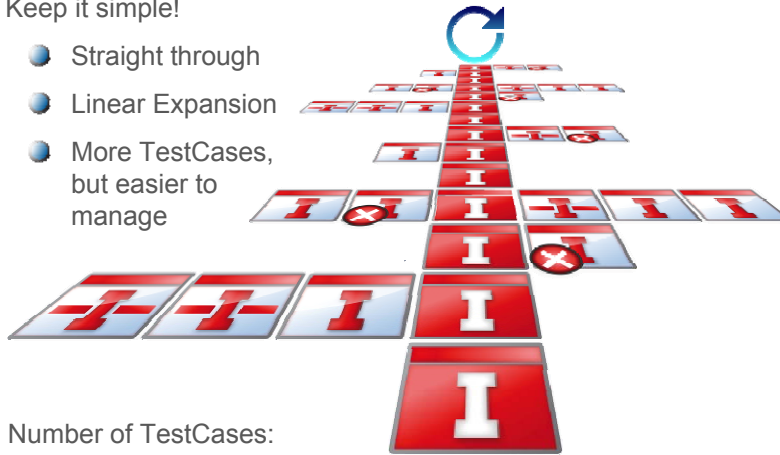
The possibility of combination

It must ideally apply to many possible instances of other attributes!



Linear Expansion

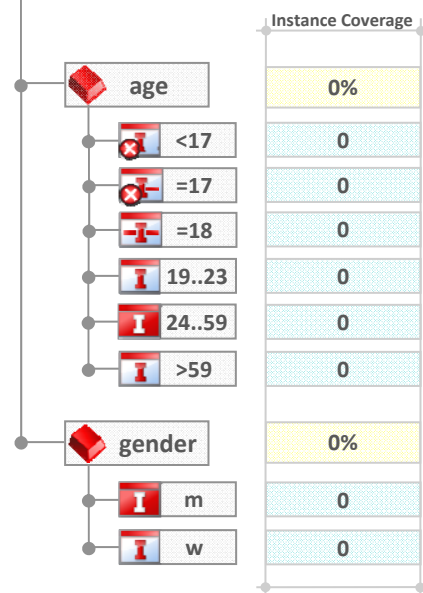
- Keep it simple!
 - Straight through
 - Linear Expansion
 - More TestCases, but easier to manage

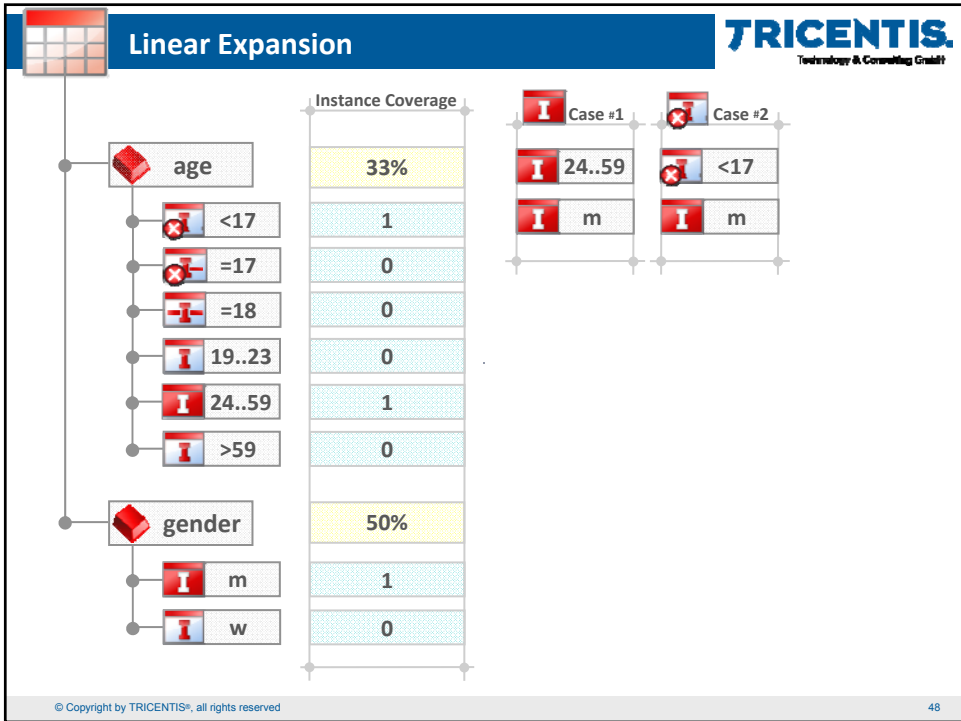
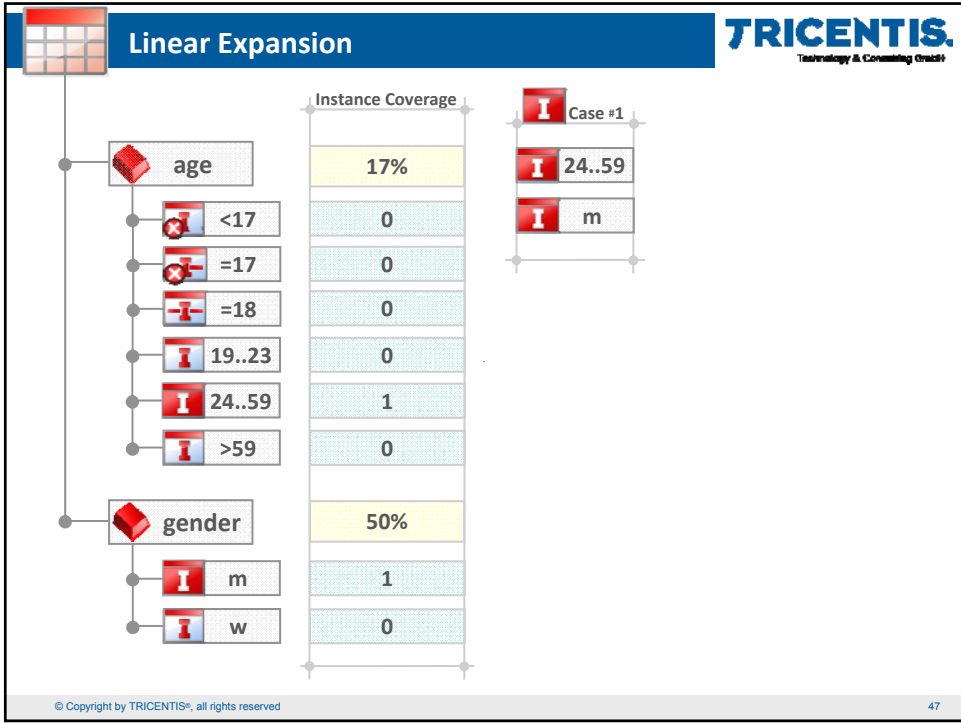


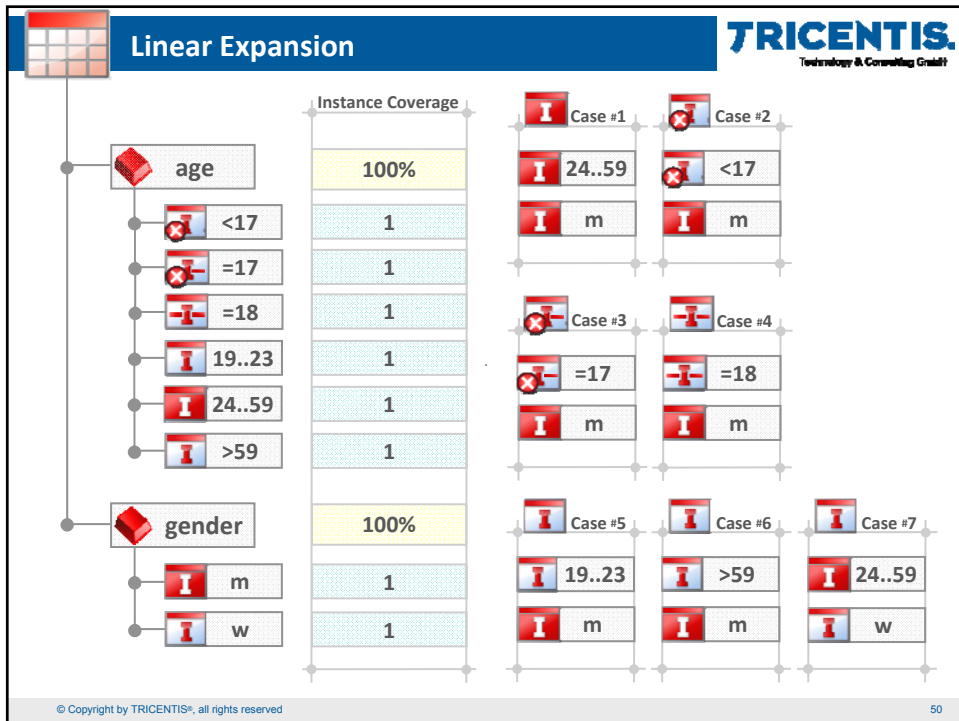
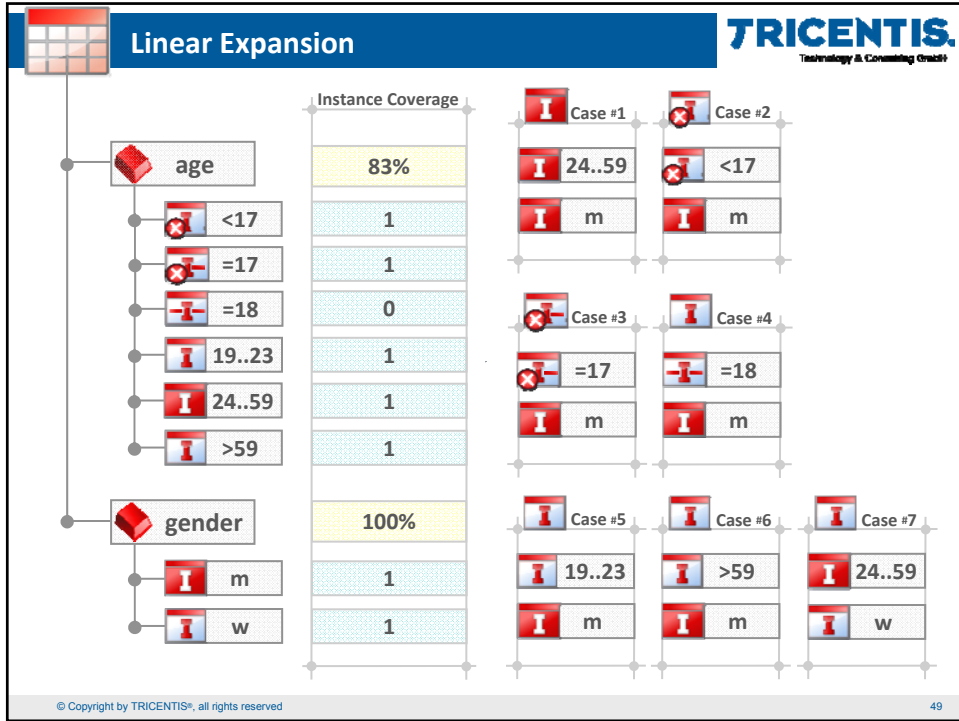
- Number of TestCases:
 - $1 + \sum (\text{Cardinalities} - 1)$
 - Cardinalities are also **summed**, not **multiplied** (thus **NO** explosion of the solution space)



Linear Expansion







Agenda

- Block I – 9:00 – 10:45 ... Problem definition & methodological basis
 - An intro to the participants and *TRICENTIS*®
 - Starting situation – presentation of the problem
 - Methodological basis of the TestCase and test data design
- <Break>
- **Block II – 11:00 – 13:00 ... TestCase and test data design**
 - Example: fictitious task from the banking sector
 - Summary

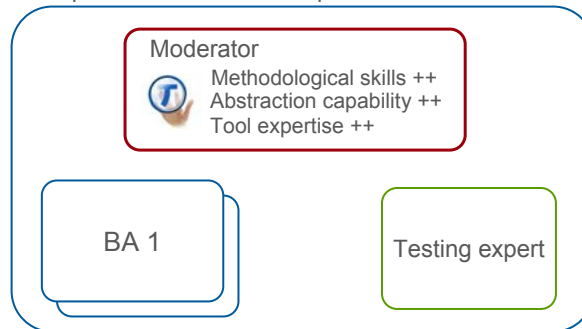


- TestCase-Design in practice (exercises in small groups, chapter 1.2 of accompanying document)
- Documentation of a sample solution in *TOSCA Testsuite™*

Workshops, which include business analysts (BA), are the most efficient types of TestCase-Design.

- Minimum 1 day and maximum 2 days en suite
- Maximum 1 workshop per week for the participants

Composition of the workshops



- Would you like to learn more about methodical testcase design?
- Would you like to elaborate and extend your knowledge even more about what you learned today and put it to work in your own environment?
- Please contact:

Ernst Jan Smit
Sales Manager Benelux
+31 6 11 533103
e.smit@tricentis.com



Thank you for your attention!