Improving Requirements Testing with Defect Taxonomies

Michael Felderer

Research Group Quality Engineering

Institute of Computer Science

University of Innsbruck, Austria

The 25th CREST Open Workshop

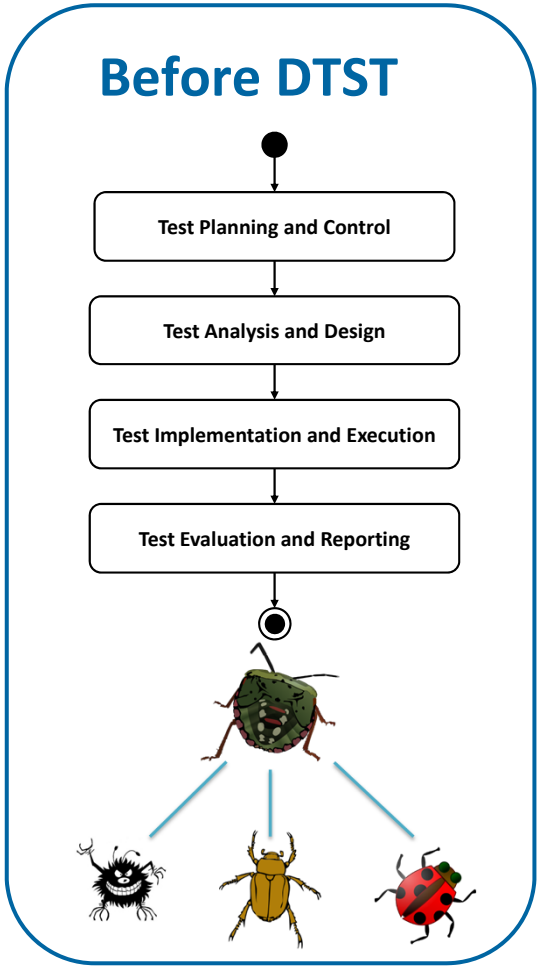
London, February 11, 2013

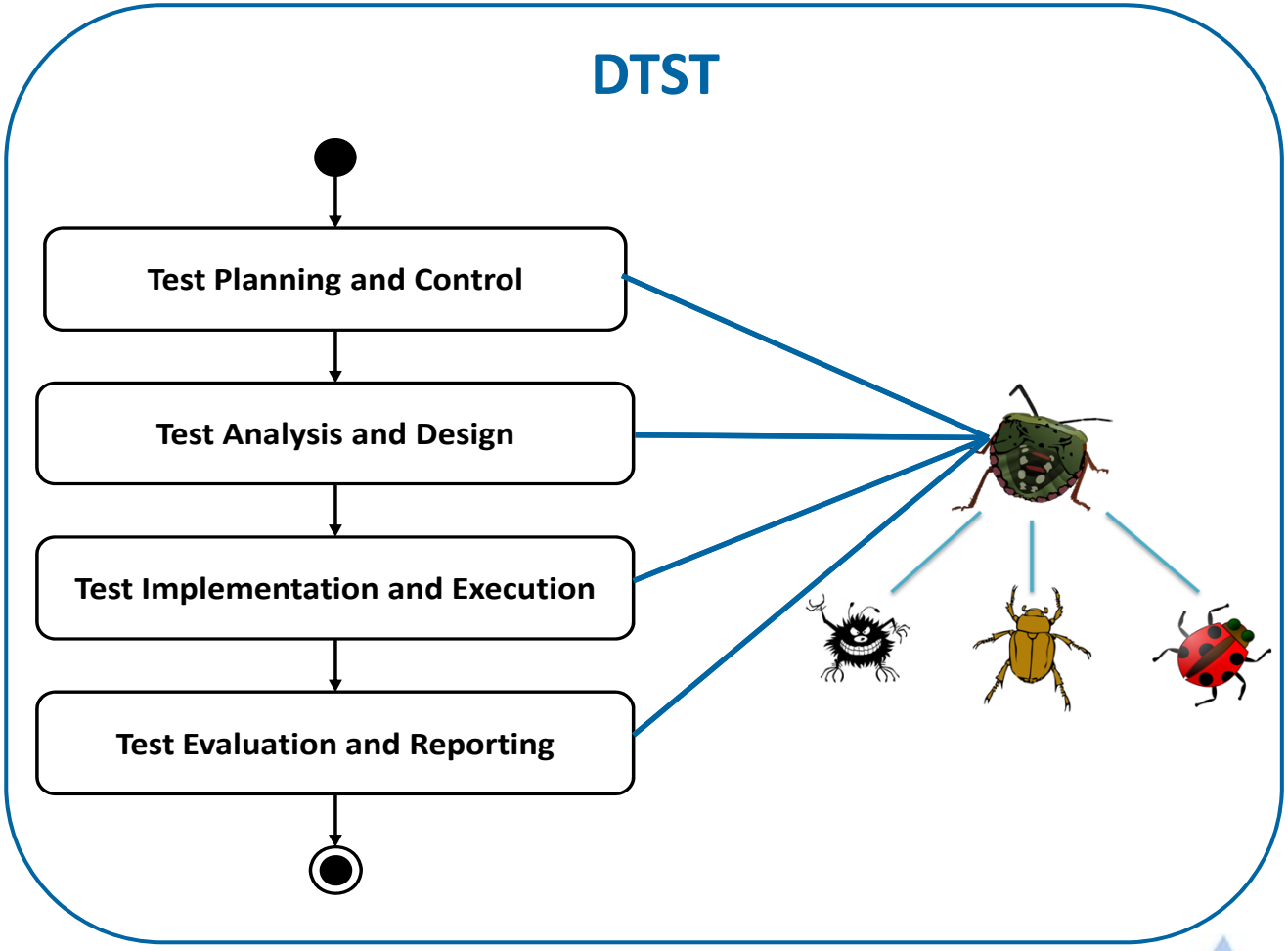
QUALITY ENGINEERING

Motivation

- Defect taxonomies (DT) provide information about distribution of failures in projects and are valuable for learning about errors being made
- DT are in practice only used for **a-posteriori allocation** of testing resources to prioritize failures for debugging purposes or for **prediction**
- But DT have potential to **control and improve overall system test process**
  - Design of requirements-based tests
  - Prioritization of test cases
  - Tracing of requirements, tests and defects
  - Controlling of defect management
  - Provide precise statement about release quality

Defect Taxonomy Supported Testing (DTST)

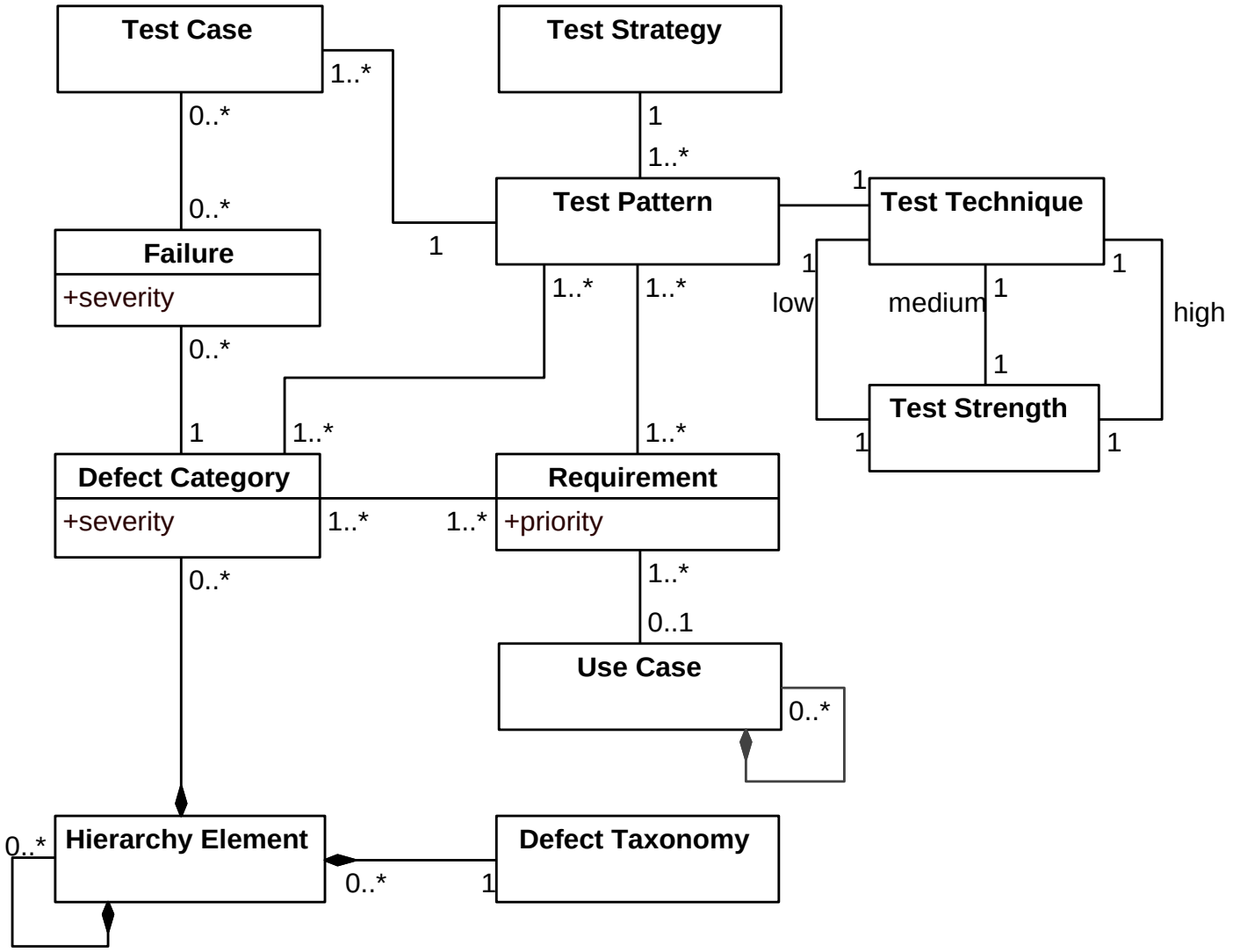




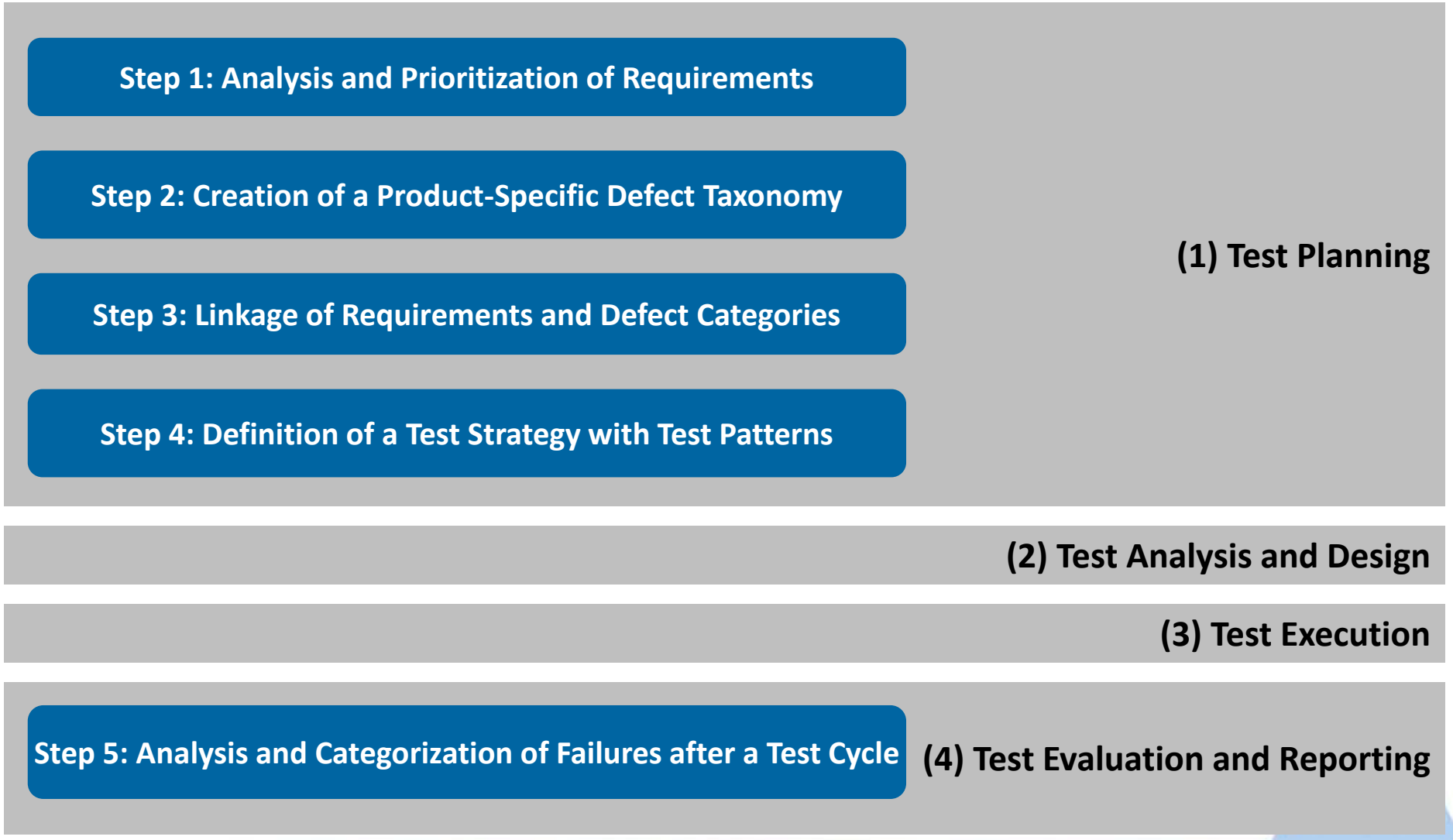
Outline

- Process for system testing with defect taxonomies, called **Defect Taxonomy-Supported Testing (DTST)**
  - Aligned with ISTQB-based standard test processes
  - Prioritized requirements, defect categories, failures
  - Traceability between requirements, defect categories and failures
  - Application of specific, goal-oriented test design techniques
  - Detailed statement about release quality
- **Empirical evaluation** of effectiveness of DTST in an industrial case study compared to standard test process
- **Decision support for the application** of defect taxonomy supported testing based on cost comparison with standard test process

Basic Concepts

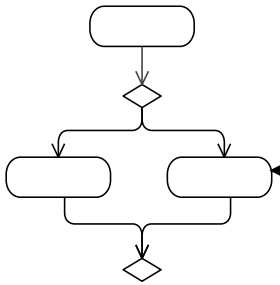


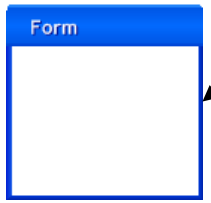
DTST Steps and Integration Standard Test Process



Step 1: Analysis and Prioritization of Requirements

- **Prioritized** requirements are **assigned to use cases** additionally defined by business processes, business rules and user interfaces





UC Identification

REQ#	Description	Priority of REQ
39	ANF_0053 Logging and Auditing	normal
40	ANF_0057 LOGIN via OID	high

UC Search Client

REQ#	Description	Priority of REQ
…	…	…

Step 2: Creation of a Product-Specific Defect Taxonomy

- Top-level categories of Beizer are mapped to **product-specific defect categories** which are then further refined to **concrete low-level defect categories** with assigned **identifier** and **severity**

Defect Category of Beizer	Product-Specific Category	DC	Description of DC	Severity
1xxx . . Requirements 11xx . . Requirements incorrect 16xx . . Requirement changes	Unsuitability of the system taking the organizational processes and procedures into account.	**R1**	Client not identified correctly	critical
		R2	Goals and measures of case manager are not processed correctly	normal
		R3	Update and termination of case incorrect	normal
12xx . . Logic 13xx . . Completeness	Incorrect handling of the syntactic or semantic constraints of GUI.	**R4**	GUI-layout Syntactic specifications of input fields Error massages	major
4xxx . Data 42xx . . Data access and handling		**D1**	Incorrect access / update of client information, states etc.	normal
		D2	Erroneous save of critical data	critical

Step 3: Linkage of Requirements and Defect Categories

- Experience-based assignment of requirements to defect categories
- Peer review of assignment important
- **Tests** are derived for **each requirement assignment**

Defect Category of Beizer	Product-Specific Category	DC	Description of DC	Severity
1xxx . . Requirements 11xx . . Requirements incorrect 16xx . . Requirement changes	Unsuitability of the system taking the organizational processes and procedures into account.	**R1**	Client not identified correctly	critical
		R2	Goals and measures of case manager are not processed correctly	normal
		R3	Update and termination of case incorrect	normal
12xx . . Logic 13xx . . Completeness	Incorrect handling of the syntactic or semantic constraints of GUI.	**R4**	GUI-layout Syntactic specifications of input fields Error massages	major
4xxx . Data 42xx . . Data access and handling		**D1**	Incorrect access / update of client information, states etc.	normal
		D2	Erroneous save of critical data	critical

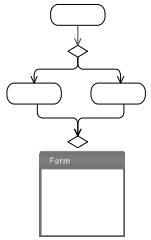
REQ#	Description	Priority of REQ
39	ANF_0053 Logging and Auditing	normal
40	ANF_0057 LOGIN via OID	high

Step 4: Definition of a Test Strategy with Test Patterns

- A test pattern consists of a **test design technique** with **three test strength** and has **assigned defect categories**

	Id	Test Design Technique	Defect Categories	Test Strength 1 (low)	Test Strength 2 (normal)	Test Strength 3 (high)
S: Sequence oriented	S1	Use case-based testing; process cycle tests	R1, R2, R3, F1,F2, F3	Main paths	Branch coverage	Loop coverage
	S3	State transition testing	I1, I2, F7, F8, F9	State coverage	State transition coverage	Path coverage
D: Data oriented	D1	CRUD (Create, Read, Update and Delete)	D1, D2		Data cycle tests	Data cycle tests
	D3	EP: Equivalence partitioning	F3, F5, F6	EP valid	EP valid+invalid	EP valid+invalid
	D4	BVA: Boundary value analysis	F3, F5, F6	BVA valid	BVA valid+invalid	BVA r values at boundaries

Test Design and Execution



Requirements

REQ	Text	PR
1	Workflow	High
2	Data	Normal

Defect Taxonomy

Defect Category of Beizer	Product-Specific Category	DC	Description of DC	Severity
1xxx . . Requirements 1[illegible]xx . . Requirements incorrect 16xx . . Requirement changes	Unsuitability of the system taking the organizational processes and procedures into account.	R1	Client not identified correctly	critical
		R2	Goals and measures of case manager are not processed correctly	normal
		R3	Update and termination of case incorrect	normal
12xx . . Logic 13xx . . Completeness	Incorrect handling of the syntactic or semantic constraints of GUI.	R4	GUI-layout Syntactic specifications of input fields Error massages	major

Test Design

	Id	Test Design Technique	Defect Categories	Test Strength 1 (low)	Test Strength 2 (normal)	Test Strength 3 (high)
S: Sequence oriented	S1	Use case-based testing; process cycle tests	R1, R2, R3, F1,F2, F3	Main paths	Branch coverage	Loop coverage
	S3	State transition testing	I1, I2, F7, F8, F9	State coverage	State transition coverage	Path coverage
D: Data oriented	D1	CRUD (Create, Read, Update and Delete)	D1, D2		Data cycle tests	Data cycle tests

PR	SDC, SF	Test Strength
high	blocker, critical, major	3
normal	blocker, critical, major	3
normal	normal, minor	2
low	minor, trivial	1

Test Execution

ID	Description	Result	Comments	Severity
1	see test spec.	pass	no	
2	see test spec.	pass	no	
3	see test spec.	fail	no	critical
4	see test spec.	pass	no	
5	see test spec.	fail	no	minor

Step 5: Analysis and Categorization of Failures after a Test Cycle

- Defects are exported from defect management tool (Bugzilla)
- Severity assigned by testers is compared to severity of defect category and priority of requirement
- Weights are adapted if needed
- Precise statement about release quality is possible
  - Additional information valuable for release planning

Case Study

- **Research Questions**

  (RQ1) Defect taxonomy-supported testing reduces the number of system test cases.

  (RQ2) Defect taxonomy-supported testing increases the number of identified system failures per system test case.

- **Evaluation via comparing**

  (RQ1) normalized number of tests (NOT/SIZE)

  (RQ2) test effectiveness (NOF/NOT)

  in **two similar projects** from a **public health insurance institution**

Studied Projects

- **Two projects** performed by the same test organization in a **public health insurance institution**

	Project A	**Project B**
Area	Application for case managers	Administration of clients of the public health insurance institution.
Staff	About 7	Up to 10
Duration	9 month development, now under maintenance	9 month development, now under maintenance
Number of iterations	4	3
SIZE: NOR + NUC	41+14	28+20
Ratio of system testing	27% of overall project effort	28% of overall project effort
Test Process	**ISTQB process + defect taxonomy-supported testing** Manual regression testing	**ISTQB process** Manual regression testing

Data Collection Procedure

1. **Requirements** stored in **Excel**, **Priorities** assigned by **Domain Experts**
2. **Defect Taxonomies** stored in **Excel**, **Severity** assigned by **Test Manager**
3. **Requirements** assigned to **Defect Categories** in **Excel**
4. **Test Cases** implemented in test management tool **TEMPPO**
5. **Defects** stored in **Bugzilla**, **Severity** assigned by **Testers**
6. **Defects** exported to **Excel** and assigned to **Defect Categories**

Results 1/2

- Collected metrics for Project A and Project B

Metrics	Project A	Project B
NOR	41	28
NUC	14	20
SIZE (NUC+NOR)	55	48
NOT	148	170
NOF	169	114
NOT/SIZE	**2.69**	**3.54**
NOF/NOT	**1.14**	**0.67**

- **NOT/SIZE indicates reduction of test cases in DTST (RQ1)**
- **NOF/NOT indicates increased test effectiveness in DTST (RQ2)**

Results 2/2

- RQ1 is additionally supported by the **estimated number of test cases without DTST** and the **actual number of test cases with DTST**

Module	NOT Estimated	NOT DTST	Reduction
GUI	32	29	9%
M1	9	8	11%
M2	58	41	29%
M3	43	34	21%
M4	14	13	7%
M5	26	23	12%
Total	**182**	**148**	**19%**

- A **paired two sample t-test** based on this table shows that the reduction of test cases is even significant (T=2.209, df=5, p=0.039)

Interpretation

- In case study we quantitatively observed that
  - DTST **reduces the number of test cases** and **increases test effectiveness**
  - Test cases are **more goal-oriented to identify failures**
  - Less resources for test implementation, execution and evaluation are needed
- Results are supported qualitatively by feedback of stakeholders
  - Testers noted that with DTST their test cases find more defects of high severity
  - Testers and managers noted less discussion on resource allocation
  - Test managers noted more realistic severity values and less unplanned releases
- Limitations of case study
  - Public health insurance domain
  - Two projects
  - Beizer defect taxonomy

Validity of Results

- Construct Validity
  - Application of standard measures on carefully selected projects
  - Data quality checks
- Internal Validity
  - Triangulation of data
    - Two independent projects
    - Consideration of estimated values
    - Qualitative feedback of stakeholders
- External Validity
  - DTST is independent of defect taxonomy and based on generic test process
  - Replication of case study in other context as future work
- Reliability
  - DTST and evaluation procedure are well-defined, data available

Cost Comparison and Decision of Application

- **Estimation** is structured according to the **phases and steps of the test process**
- **Comparison of cost** (measured in time) of **DTST** and **standard ISTQB-based test process**
- **Break-even** and recommendation of application if cost of DTST is smaller than cost of ISTQB
- Return on Investment provides same qualitative result
- Adaptation and **simulation with parameters**
  - Analysis and prioritization of a requirement
  - Linkage of a requirement to defect category
  - Design and implementation of a test case
  - Execution time per test case
  - Maintenance time per test case

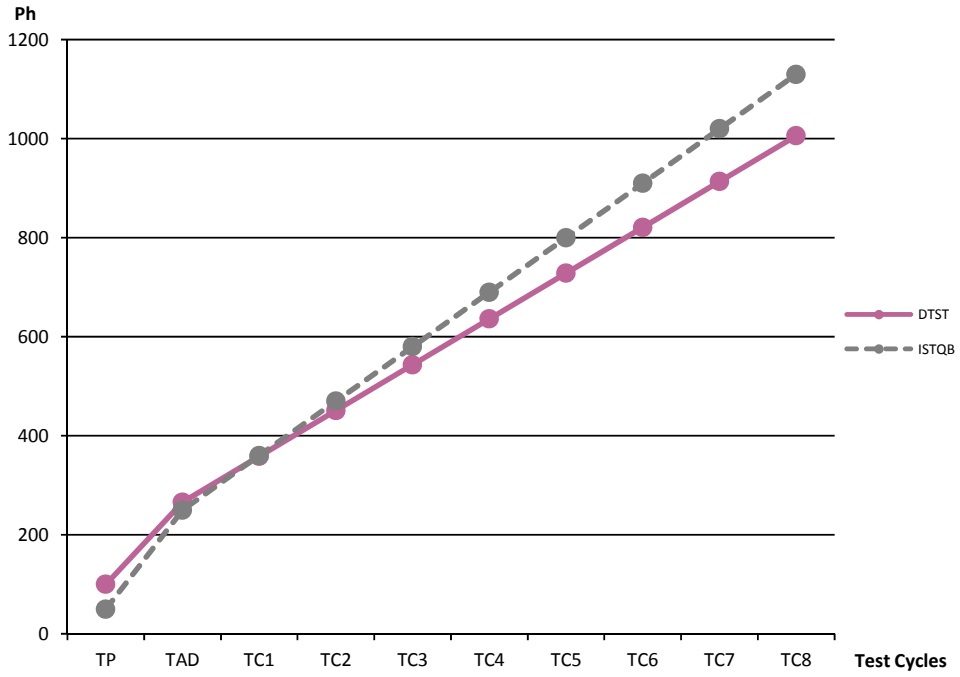
Cost Comparison for Project A

	ISTQB	DTST	Diff of Costs
Phases	**Ph**	**Ph**	**Ph**
TE Test Planning	**50.00**	**100.00**	**-50.00**

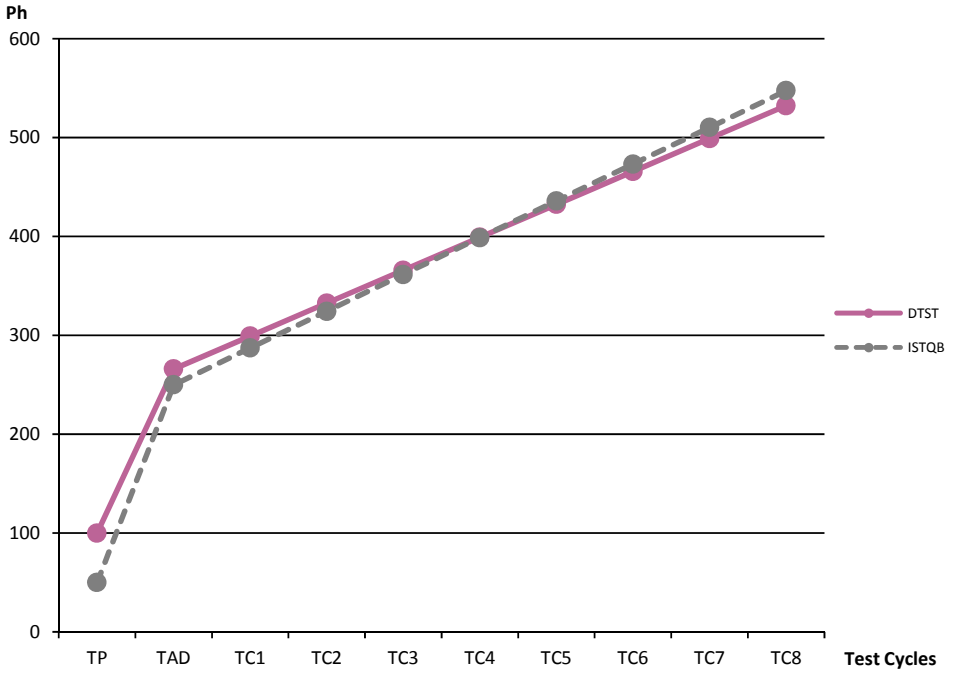
Activities	**Ph**	**Activities**	**Ph**
Analysis and prioritization of requirements **40** requirements @ **0.5 Ph**	20,00	Step 1: Analysis and prioritization of requirements **40** requirements @ **0.5 Ph**	20,00
		Step2: Creation of a product-specific defect taxonomy	30,00
		Step 3: Linkage of requirements and defect categories **40** requirements @ **0.25 Ph**	10,00
Definition of test strategy	30,00	Step 4: Definition of a test strategy with test patterns	40,00
	50,00		**100,00**

TC6	**910.00**	**821.00**	**89.00**
TC7	**1020.00**	**913.50**	**106.50**
TC8	**1130.00**	**1006.00**	**124.00**

Cost Comparison Scenarios for Project A



Cost comparison with average test execution time **0.5h**



Cost comparison with average test execution time **0.1h**

Summary and Future Work

- **Summary**
  - **Standard aligned** defect taxonomy supported system test process
  - Industrial case study indicates **more effective test cases**
  - **Procedure for deciding** whether to apply the approach based on **cost comparison**
- **Future work**
  - Impact of **different defect taxonomy types**
  - **Automatic recommendations** for assignments
  - **Requirements validation** with defect taxonomies
  - **Regression testing** with defect taxonomies
  - Application of defect taxonomy supported testing for **release planning**
  - Investigate the **value** contribution of defect taxonomy-supported testing, e.g. reduced release planning uncertainty, severe bugs in operation

Questions or Suggestions ?

michael.felderer@uibk.ac.at