The Value of Testing in 5 Dimensions

OOP 2011
Munich, Germany

Peter Zimmerer
Principal Engineer
Siemens AG
Corporate Technology
Corporate Research and Technologies
System Development Technologies
81739 Munich, Germany
peter.zimmerer@siemens.com
http://www.siemens.com/corporate-technology/

Contents

SIEMENS

Some *old* definitions

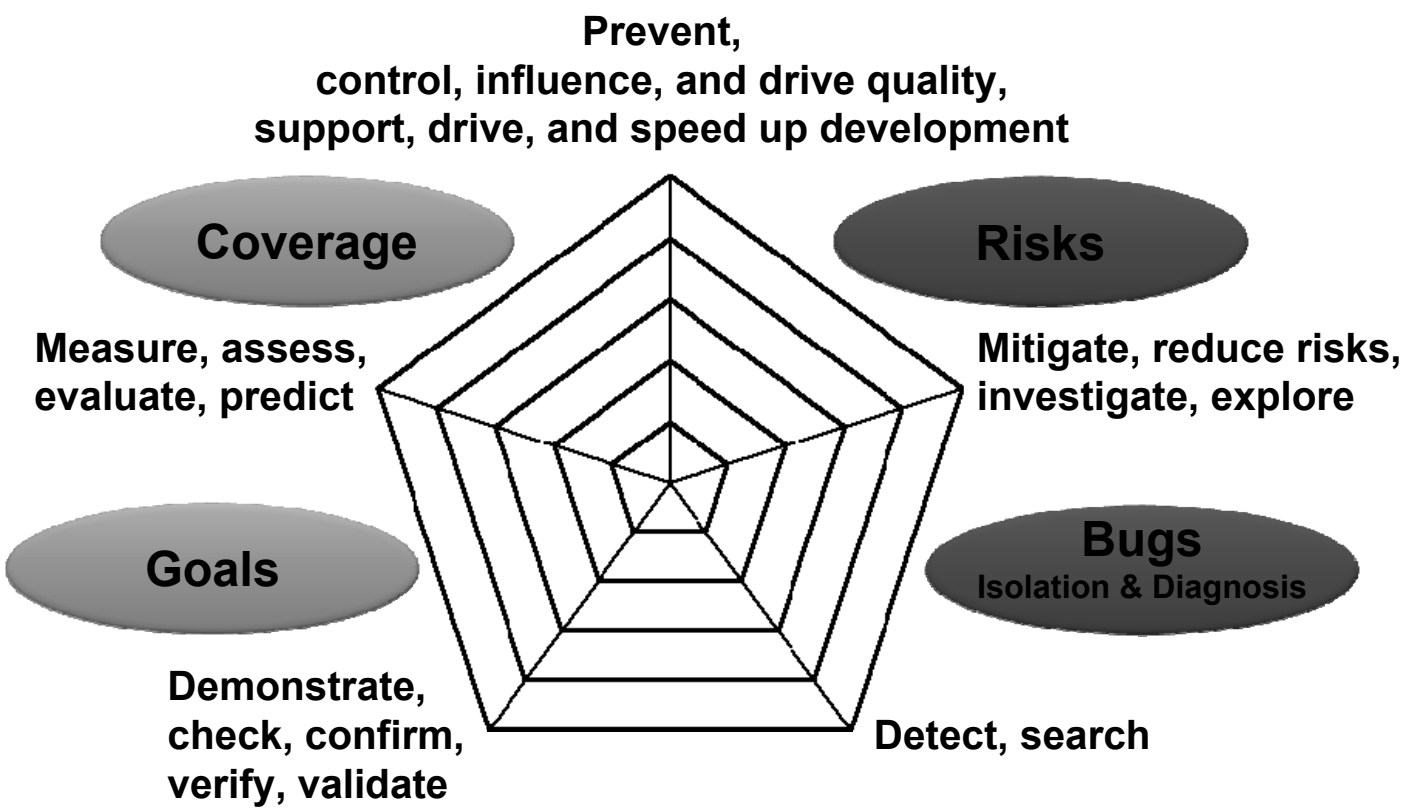
IEEE Standard 610.12–1990
The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component.

IEEE Standard 829–1998
The process of analyzing a software item to detect the differences between existing and required conditions (that is, bugs) and to evaluate the features of the software item.

BCS SIGIST Testing Standards Working Party (http://www.testingstandards.co.uk/living_glossary.htm)
The process of exercising software to verify that it satisfies specified requirements and to detect errors.

SIEMENS

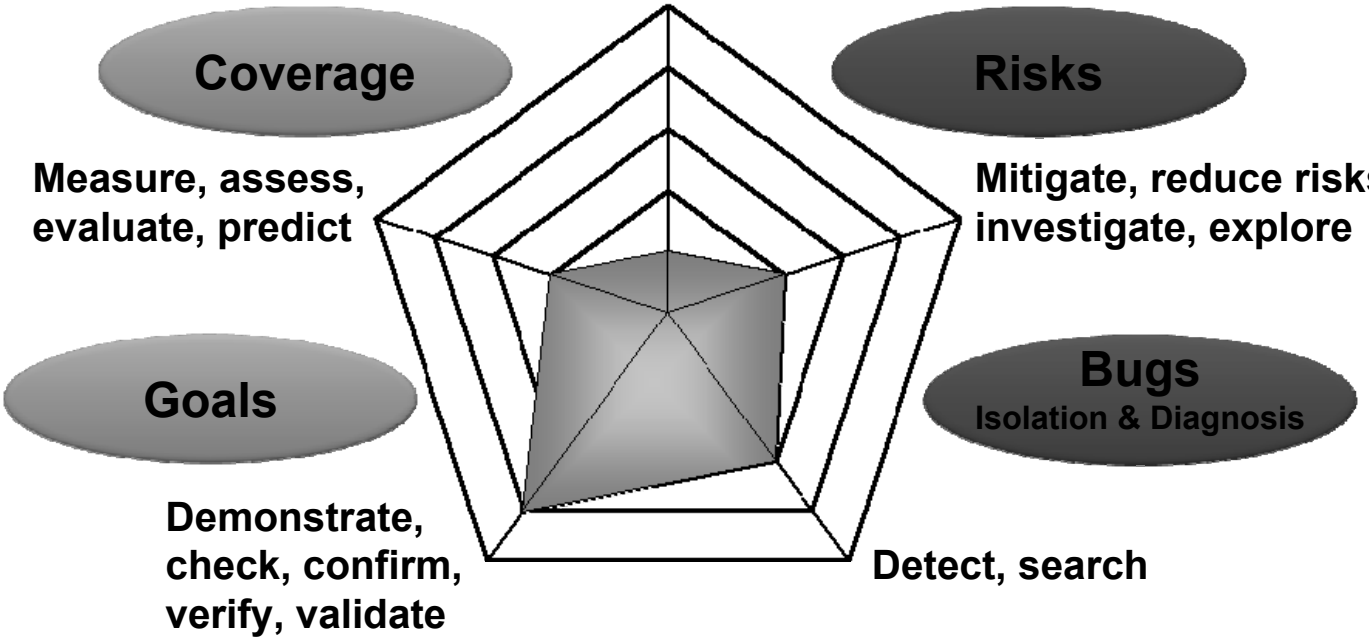
Why do we test? Dimensions of testing



SIEMENS

Why do we test? Dimensions of testing

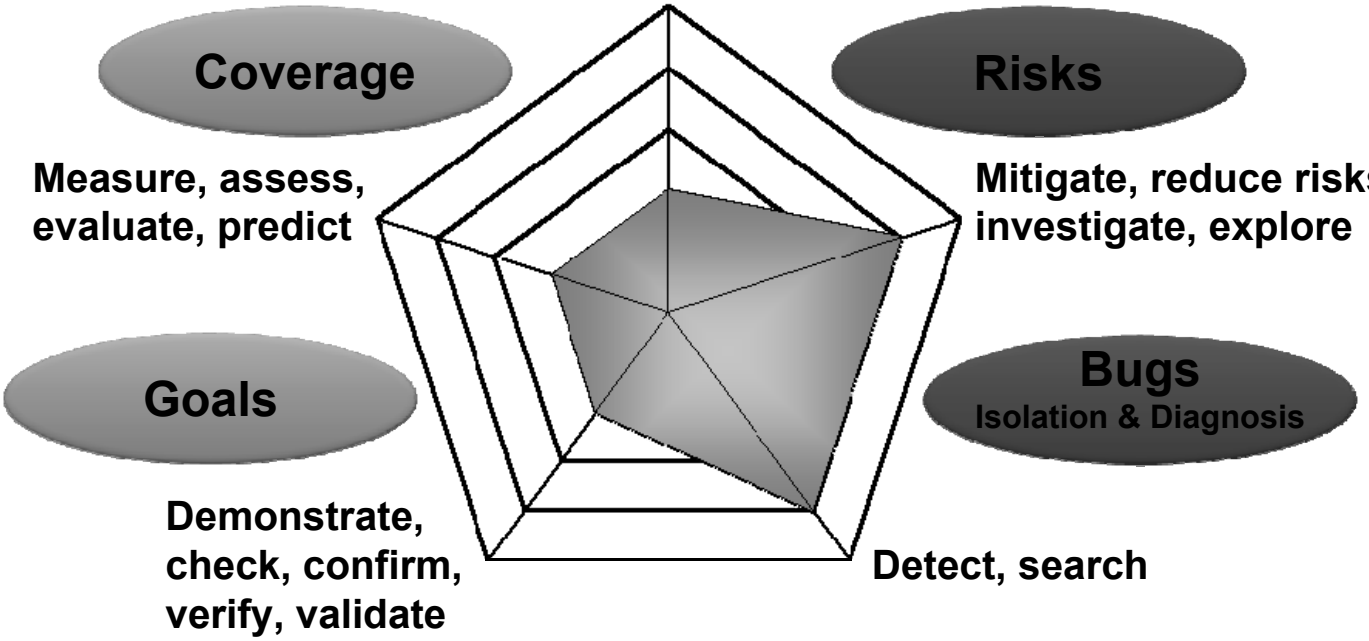
Prevent,
control, influence, and drive quality,
support, drive, and speed up development



SIEMENS

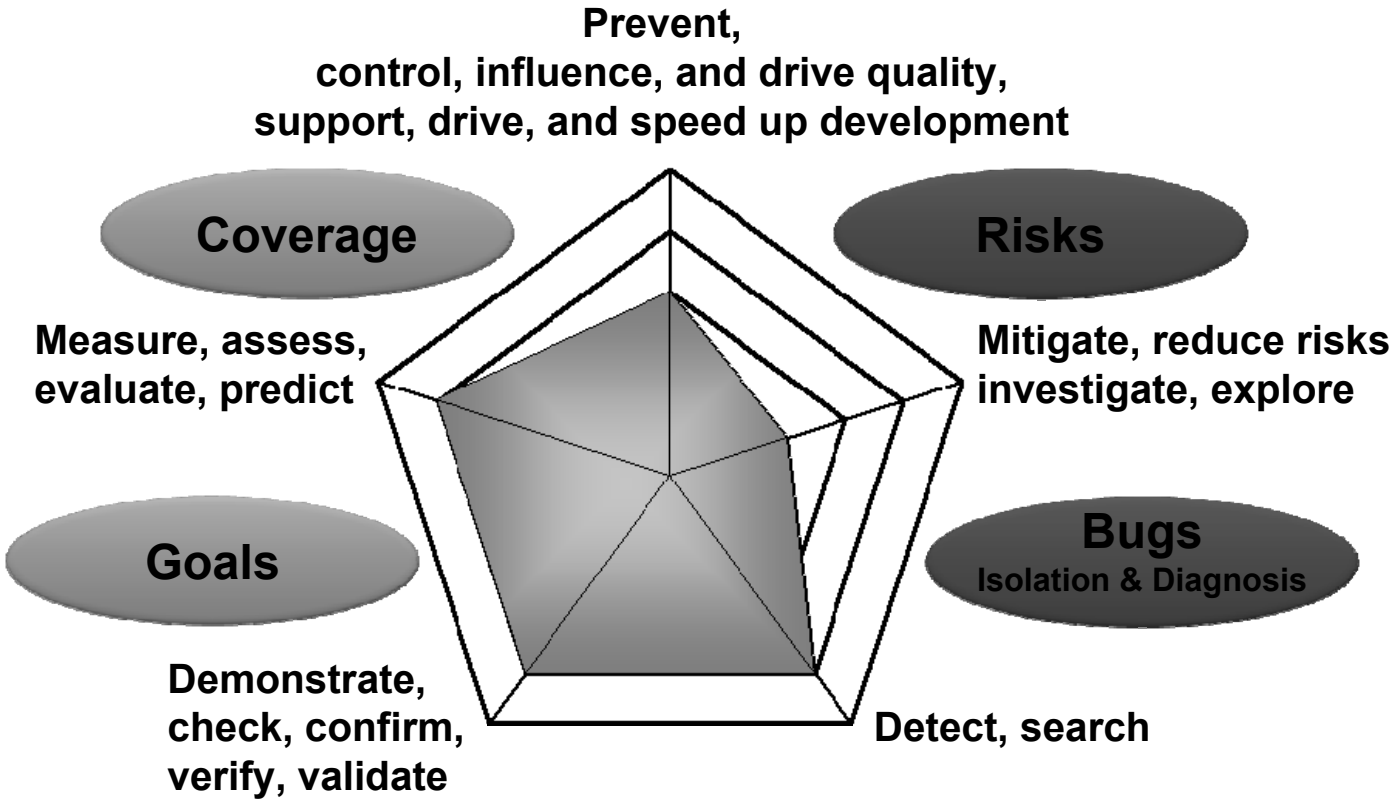
Why do we test? Dimensions of testing

Prevent,
control, influence, and drive quality,
support, drive, and speed up development



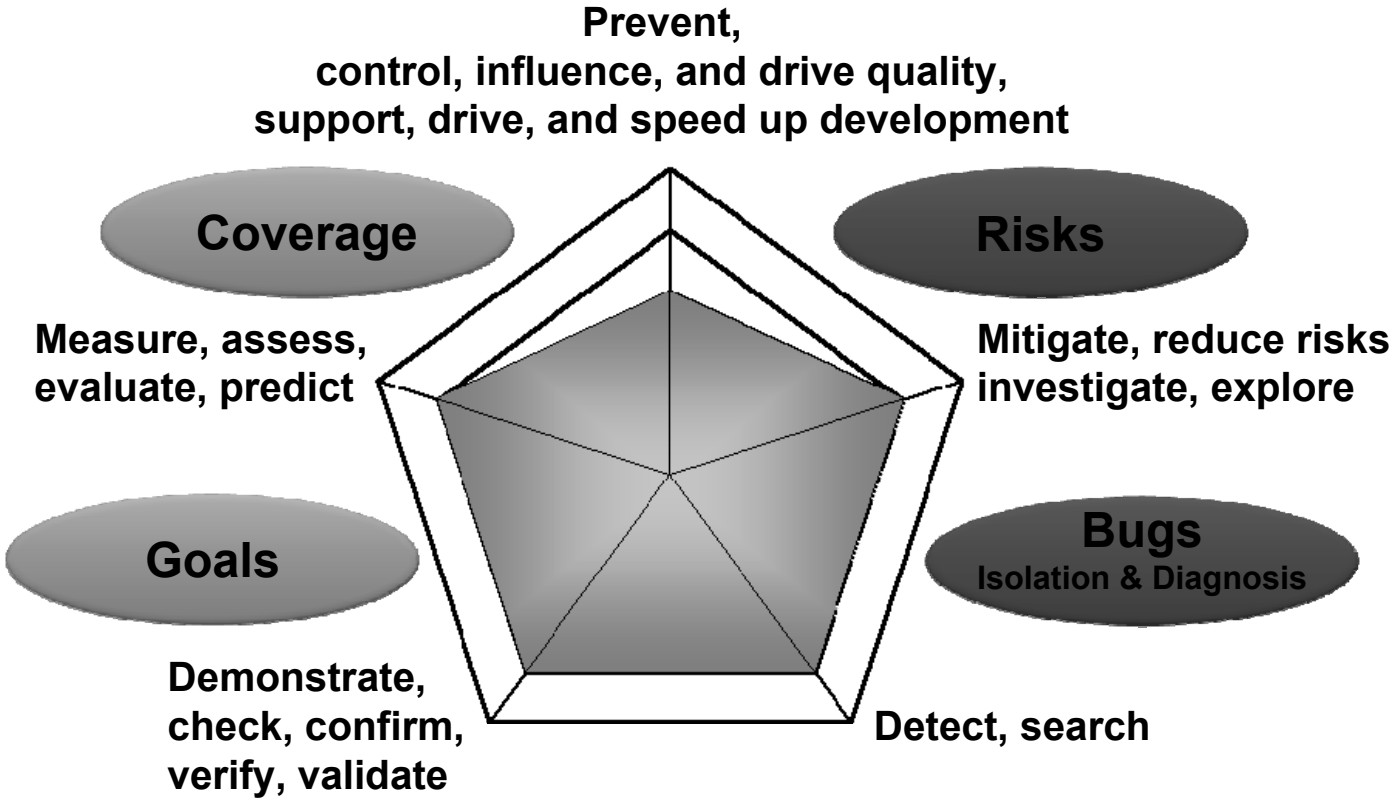
SIEMENS

Why do we test? Dimensions of testing

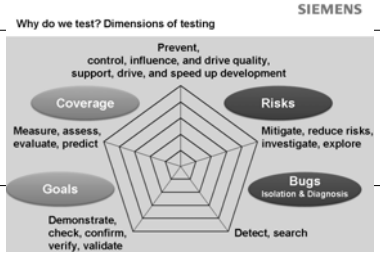


SIEMENS

Why do we test? Dimensions of testing



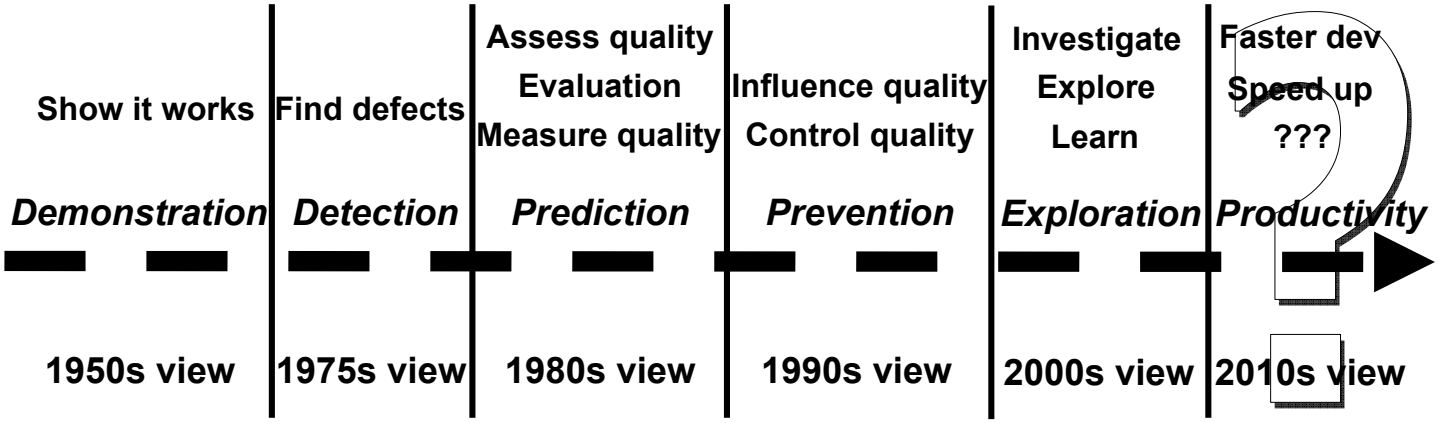
Why do we test?



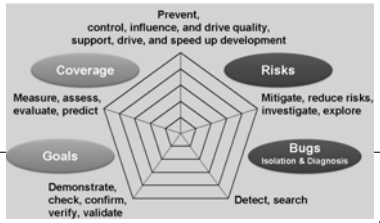
SIEMENS

Historical and future (?) view

History consists of a series of accumulated imaginative inventions.
Voltaire, 1694–1778



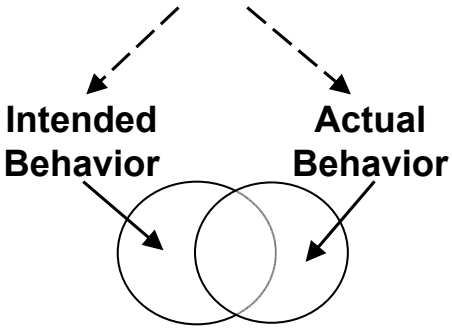
Why do we test?
What is the *value* of testing?



SIEMENS

Empirical technical investigation of the product / system / artifact / service under test conducted to provide stakeholders with information about the quality.
Cem Kaner

- **Product**
- **Decisions**
- **Process**



If this information is effectively used, then we create real value for the business, i.e. the business value of testing lies in the savings that the organization can achieve from improvements based on the information that is provided by testing.

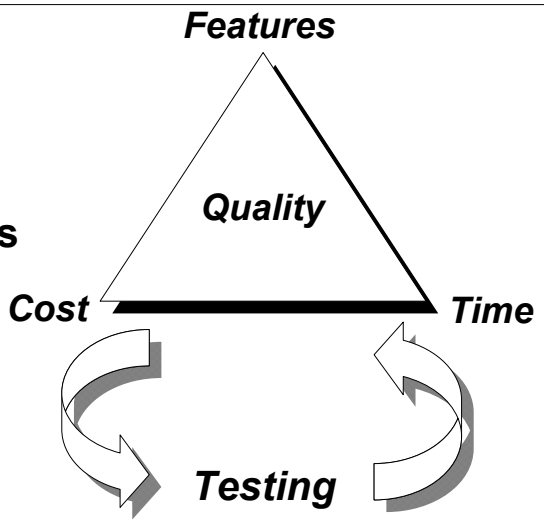
More / better testing means more / better information and evidence!

SIEMENS

Why do we test? What is the *value* of testing?

We do testing
to provide *information* and *evidence*

The value of *information* and *evidence* is for stakeholders to decide



Features, cost, and time are inputs only
→ Only testing can measure achievement in a project
→ Testing is the ONLY source of *information* and the ONLY source of *evidence* we have to make sensible decisions on availability and readiness

SIEMENS

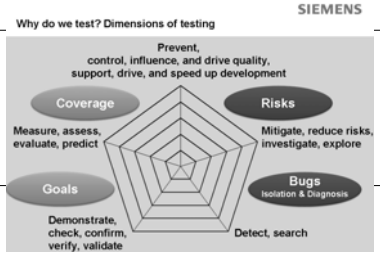
Some *better* definitions

Cem Kaner
Empirical technical investigation of the product / system / service under test conducted to provide stakeholders with information about the quality.

Software Quality Engineering (SQE)
All lifecycle activities concerned with checking software and software-related work products. Testing is an activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. Testing includes all activities associated with the planning, preparation, execution, and reporting of tests.

ISTQB Glossary (2007) (http://www.istqb.org/)
The process consisting of all life cycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

Addressing typical questions and challenges in testing



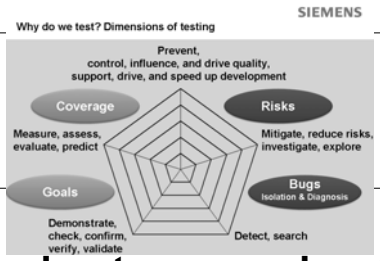
SIEMENS

Developer:
My unit testing does not detect any important new bugs. Why should I do unit testing in the future?

Manager:
You have this big regression test suite. Why do we need to spend so much effort on just repeating these tests again and again?

Project Lead:
You have all this great test automation in place, but the system still crashes quite often. What shall we do?

FAQ



SIEMENS

- **Why are there just 5 dimensions and not more or less?**
- **Don’t some dimensions contain too many different things?**
- **Are the different dimensions really disjoint?**
- **The figure looks like a kiviat diagram. Is that intended?**
- **Is there any semantics in the ordering of the dimensions in the figure?**

Risk-based testing strategy

Base the testing strategy on business goals and priorities
→ *Risk-based testing strategy (RBT)*

Methods, paradigms, techniques, styles, and ideas to create, derive, select, generate a test case

Risk identification

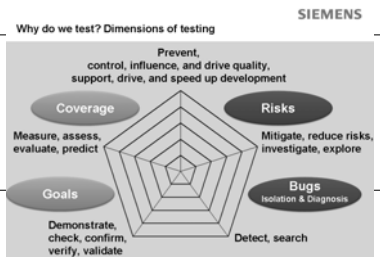
Risk = P × D

- **P Probability of failure**
  - Frequency of use
  - Chance of failure:
    criticality & complexity at implementation & usage, lack of quality
- **D Damage (consequence & cost for business & test & usage)**

Risk analysis – Product risk analysis workshop

Risk response – Test objectives, test levels, test design methods …

Categories of test design methods



Regression

Positive, valid

Negative, invalid

Grey-box

Fault-based

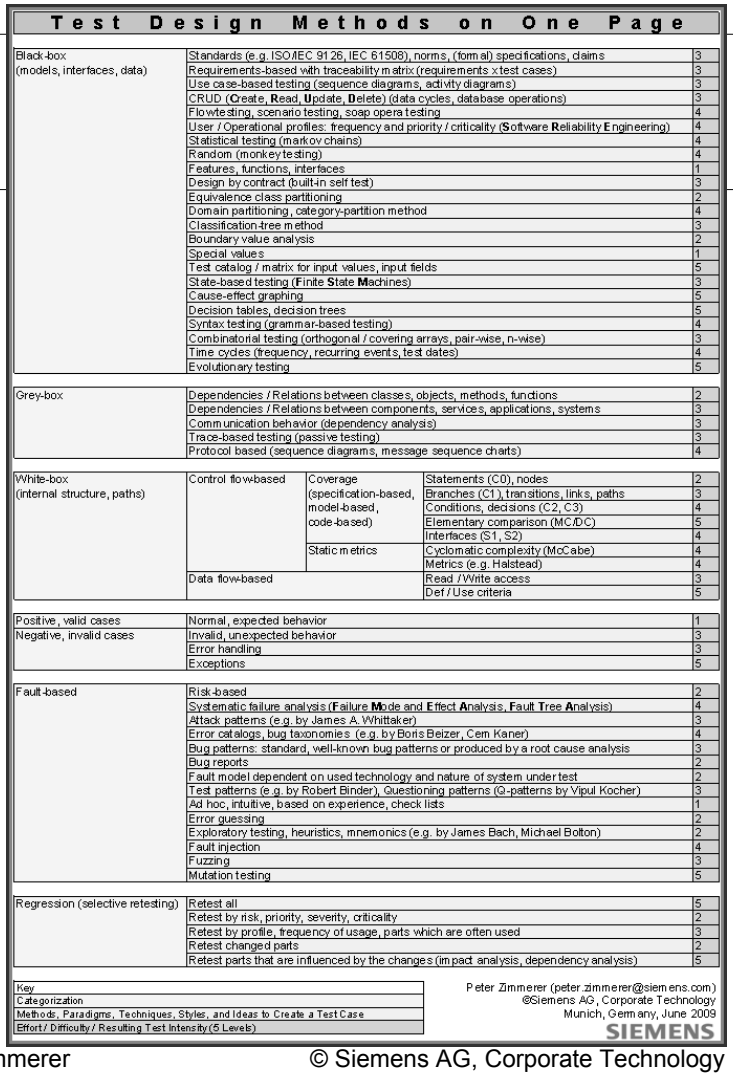
Models, interfaces, data

Internal structure, paths

Black-box

White-box: control flow, data flow

Poster *Test Design Methods on One Page*



Test Design Methods on One Page

Black-box (models, interfaces, data)	Standards (e.g. ISO/IEC 9126, IEC 61508), norms, (formal) specifications, claims			3
	Requirements-based with traceability matrix (requirements x test cases)			3
	Use case-based testing (sequence diagrams, activity diagrams)			3
	CRUD (**C**reate, **R**ead, **U**pdate, **D**elete) (data cycles, database operations)			3
	Flow testing, scenario testing, soap opera testing			4
	User / Operational profiles: frequency and priority / criticality (**S**oftware **R**eliability **E**ngineering)			4
	Statistical testing (markov chains)			4
	Random (monkey testing)			4
	Features, functions, interfaces			1
	Design by contract (built-in self test)			3
	Equivalence class partitioning			2
	Domain partitioning, category-partition method			4
	Classification-tree method			3
	Boundary value analysis			2
	Special values			1
	Test catalog / matrix for input values, input fields			5
	State-based testing (**F**inite **S**tate **M**achines)			3
	Cause-effect graphing			5
	Decision tables, decision trees			5
	Syntax testing (grammar-based testing)			4
	Combinatorial testing (orthogonal / covering arrays, pair-wise, n-wise)			3
	Time cycles (frequency, recurring events, test dates)			4
	Evolutionary testing			5
Grey-box	Dependencies / Relations between classes, objects, methods, functions			2
	Dependencies / Relations between components, services, applications, systems			3
	Communication behavior (dependency analysis)			3
	Trace-based testing (passive testing)			3
	Protocol based (sequence diagrams, message sequence charts)			4
White-box (internal structure, paths)	Control flow-based	Coverage (specification-based, model-based, code-based)	Statements (C0), nodes	2
			Branches (C1), transitions, links, paths	3
			Conditions, decisions (C2, C3)	4
			Elementary comparison (MC/DC)	5
			Interfaces (S1, S2)	4
		Static metrics	Cyclomatic complexity (McCabe)	4
			Metrics (e.g. Halstead)	4
	Data flow-based		Read / Write access	3
			Def / Use criteria	5
Positive, valid cases	Normal, expected behavior			1
Negative, invalid cases	Invalid, unexpected behavior			3
	Error handling			3
	Exceptions			5
Fault-based	Risk-based			2
	Systematic failure analysis (**F**ailure **M**ode and **E**ffect **A**nalysis, **F**ault **T**ree **A**nalysis)			4
	Attack patterns (e.g. by James A. Whittaker)			3
	Error catalogs, bug taxonomies (e.g. by Boris Beizer, Cem Kaner)			4
	Bug patterns: standard, well-known bug patterns or produced by a root cause analysis			3
	Bug reports			2
	Fault model dependent on used technology and nature of system under test			2
	Test patterns (e.g. by Robert Binder), Questioning patterns (Q-patterns by Vipul Kocher)			3
	Ad hoc, intuitive, based on experience, check lists			1
	Error guessing			2
	Exploratory testing, heuristics, mnemonics (e.g. by James Bach, Michael Bolton)			2
	Fault injection			4
	Fuzzing			3
	Mutation testing			5
Regression (selective retesting)	Retest all			5
	Retest by risk, priority, severity, criticality			2
	Retest by profile, frequency of usage, parts which are often used			3
	Retest changed parts			2
	Retest parts that are influenced by the changes (impact analysis, dependency analysis)			5

Key
Categorization
Methods, Paradigms, Techniques, Styles, and Ideas to Create a Test Case
Effort / Difficulty / Resulting Test Intensity (5 Levels)

Peter Zimmerer (peter.zimmerer@siemens.com)
©Siemens AG, Corporate Technology
Munich, Germany, June 2009
SIEMENS

SIEMENS

Test design methods – Some examples

Black-box
Requirements-based with traceability matrix
Use case-based testing
Scenario testing

Equivalence class partitioning
Boundary value analysis

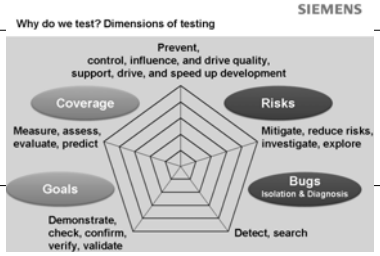
State-based testing

Grey-box Dependencies, communication behavior, protocol based

White-box Control flow (branches, transitions) and data flow

Fault-based
Risk-based
Fault model dependent on used technology

Summary



SIEMENS

Characterize testing in 5 dimensions to visualize its mission, motivations, activities, and corresponding values

Use it

- **to communicate the testing mission and its value in an appropriate manner to the different stakeholders**
- **as starting point to define the testing strategy**

Identifying the required intensity for each dimension and selecting an adequate mixture of them to cover the testing space are the key for effective and efficient testing

This will result in better collaboration with the other stakeholder, and it will help us also to strengthen and sustain our own testing territory

