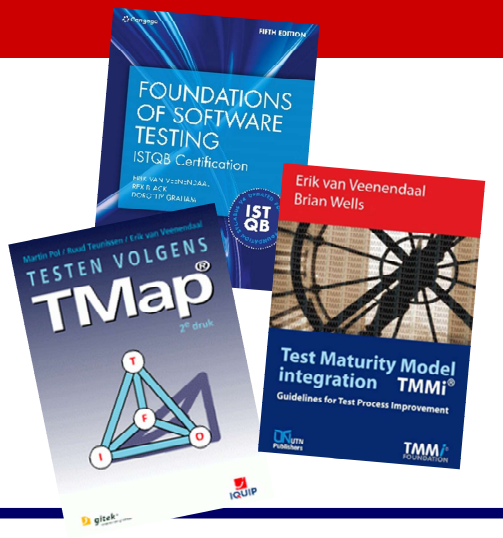
Building on Success in the AI era

Erik van Veenendaal
www.erikvanveenendaal.nl



Test Maturity

Majority is still TMMi Level 1

LEVEL 5: OPTIMIZATION

- Defect Prevention
- Test Process Optimization
- Quality Control

LEVEL 4: MEASURED

- Test Measurement
- Product Quality Evaluation
- Advanced Reviews

LEVEL 3: DEFINED

- Test Organization
- Test Training Program
- Test Lifecycle and Integration
- Non-Functional Testing
- Peer Reviews

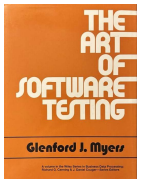
LEVEL 2: MANAGED

- Test Policy and Strategy
- Test Planning
- Test Monitoring and Control
- Test Design and Execution
- Test Environment

LEVEL 1: INITIAL

Test Policy and Strategy
Test Planning
Test Monitoring and Control
Test Design and Execution
Test Environment

Core Practice # 1: Test Strategy

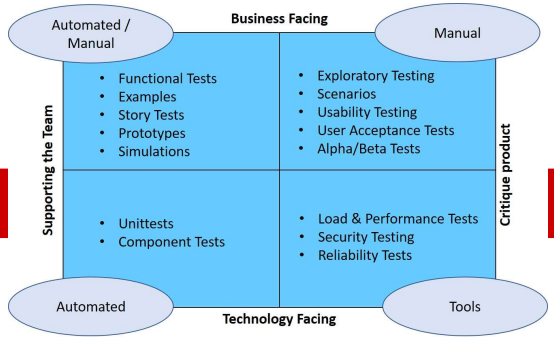


Testing must be planned

Define your Test Levels / Types / Quadrants

- Understand your Testing
- Who does What, How and When
- Don't assume, Less Overlap, No Holes

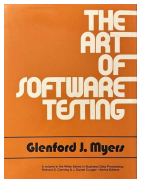
Test Strategy



Test level	Definition	Objective	Responsibility	DoR	DoD	Deliverables	Environment	Test technique	NF
Unit test									
Contract test									
System test									
SIT									
End-to-End test									
UAT									

Test Type	Goal	Responsibility	Entry criteria / Definition-of-ready	Exit criteria / Definition-of-done	Deliverables	Environment	Non-functional & other test types	Test techniques
Unit test	The unit test must demonstrate that the developed code works technically and functionally (according to design).	Developers (supported by test analysts within the team)	Code unit is developed and can be compiled. Code review (including rework) completed.	Unit test passes and SonarQube statement coverage is > 85%. Code review done.	Unit test reports. Automated Unit tests.	Localhost		Statement testing, reviews
Functional test	Demonstrate that the new/modified functionality works according to the requirements (user stories). Ensure system is set up correctly (test configuration)	BA/ Test analysts within the team. Developers.	1. Functional design or user stories 2. Test environment available. New/modified software available.	1. Software meets acceptance criteria. 2. No priority items or defects (or risks covered/accepted) open.	Test cases, results	Test / QA	Usability checks	Scenario-based, positive & negative
System integration test	Ensure full business process works.	IT Test Team	All components integrated	Business flow works end-to-end.	Execution report	Test / QA	Stability checks	End-to-end workflows
User acceptance test	Validate that the system meets business requirements and supports real user processes before go-live.	Business users, Product Owners, supported by Test Analysts.	Clear business requirements and acceptance criteria are defined. Test scenarios agreed with stakeholders. Stable build deployed in UAT environment.	Business sign-off	Signed UAT	UAT	Usability & operational checks	Business scenarios
Non-Functional test	Ensure the system performs under expected and peak loads with acceptable response times, throughput, and stability.	Performance Test Engineers, supported by Infrastructure/DevOps team.	Test environment replicates production scale. Workload model and performance scenarios defined. Monitoring tools configured.	System meets agreed performance SLAs (response times, throughput, error rates). No performance bottlenecks remain unresolved.	Performance test report (metrics, graphs, analysis). Identified bottlenecks with recommendations.	Dedicated performance test environment mirroring production.	Non-functional testing (load, stress, soak, scalability).	Load testing, stress testing, endurance testing, volume testing.
Security test	Validate that the system is protected against vulnerabilities, unauthorized access, and data breaches.	Security specialists / Pen-testers, supported by developers.	Security requirements and compliance criteria defined. Test scope and tools agreed. Stable build available.	No critical/high vulnerabilities remain. All mandatory security checks pass. Risks documented and accepted by stakeholders.	Security test report, vulnerability log, remediation actions.	Security test environment (preferably production-like).	Non-functional testing (penetration, vulnerability, authentication, authorization)	Penetration testing, vulnerability scanning, risk-based testing, ethical hacking techniques.
Sytem / End-to-End Testing	Ensure full business process works.	IT Test Team	All components integrated	Business flow works end-to-end.	Execution report	Test / QA	Stability checks	End-to-end workflows

Core Practice # 2: Risk-Based



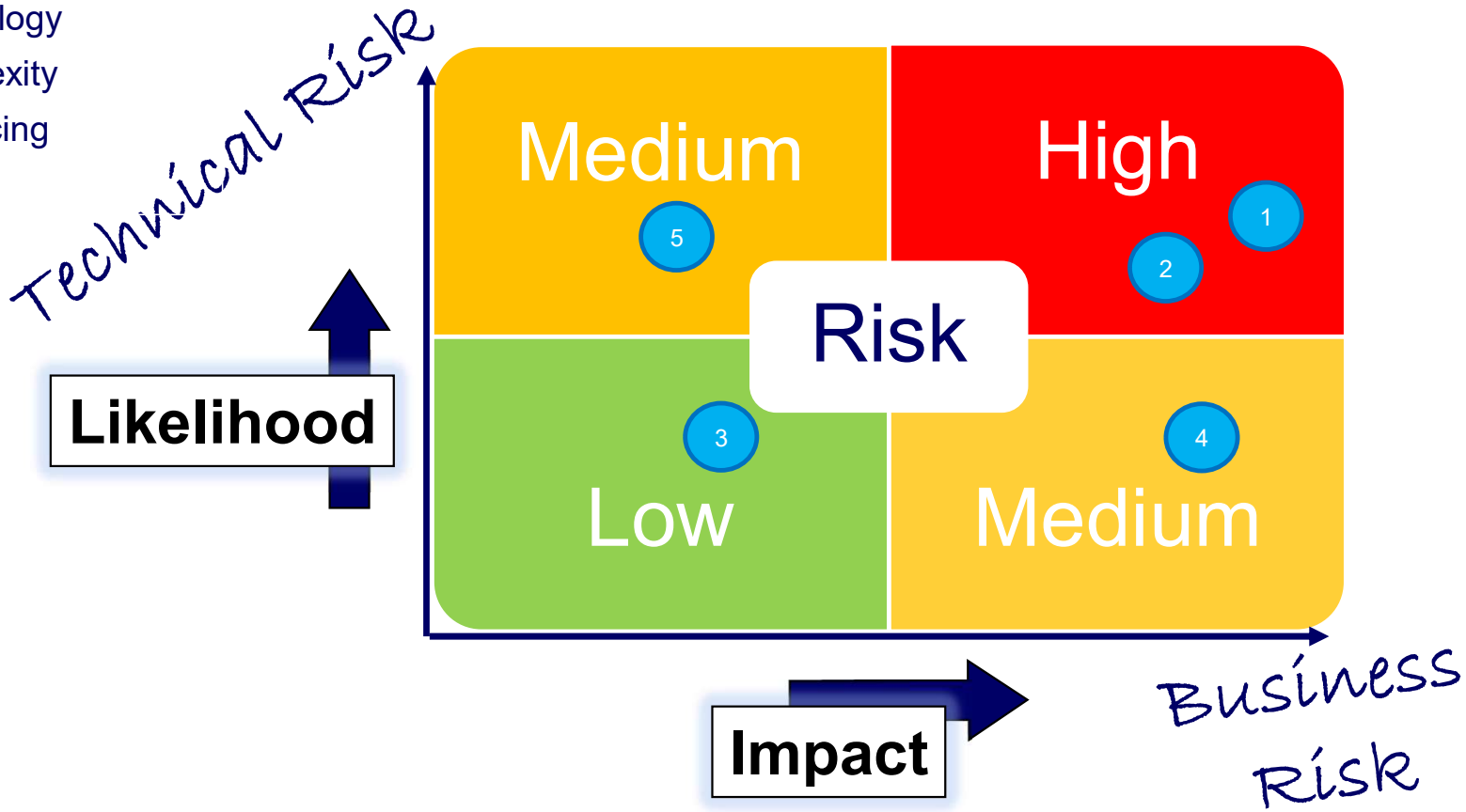
Testing is risk-based

Define Clear Testing Priorities

- Product Risk Session (refinements)
- Risk-Based Test Approach
- Quality is a Team Responsibility

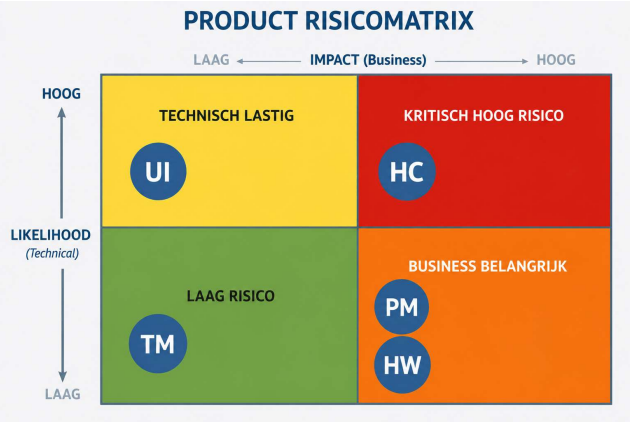
Product Risk Matrix

- Technology
- Complexity
- Interfacing
- …



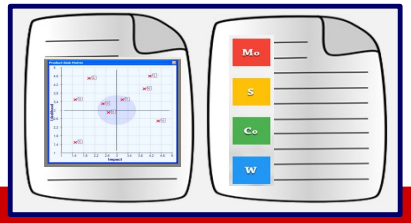
- Safety
- Usage intensity
- External visibility
- …

GenAI



User Story	Likelihood of Defects	Rationale (Likelihood)	Impact of Defects	Rationale (Impact)	Quadrant
UI	High	Completely new UI Design, high complexity, high LOC, major selling feature, usability risks, developed by experienced engineer but largely new functionality	High	Core-customer facing functionality, poor UX directly impacts product attractiveness and usability; high usage frequency	I
PM	Medium	Mostly re-used functionality, moderate complexity, medium LOC, assigned partly to junior engineer, increasing implementation risk	High	Essential operational functionality, inability to power on/off renders machine unusable, affects all users	III
HC	Low-Medium	Very low complexity, small LOC, heavily re-used, implemented by experienced engineer	Very High	Safety critical functionality, failure may cause overheating or fire hazard, severe manufacturer and customer impact	III
TM	High	New functionality added, moderate complexity, handled by experienced engineer, timing/schedule logic often defect-prone	Medium	Convenience feature only, lower usage frequency, failure does not stop core coffee-machine functionality	II
HW	Medium	Existing drivers re-used, but hardware integration always carries technical risk, relatively high complexity, hardware/software integration risks remain	High	Failure may prevent communication with physical machine components, impacts core functionality	III

Test Approach



Test design techniques
Positive tests in regression
Code review
100% code coverage

Test design techniques / ET
Review system test cases
All tests in regression set
Code review
100% code coverage

Risk

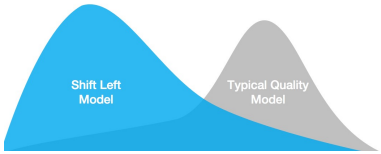
Likelihood

Impact

Exploratory testing
Pair review
85% code coverage

Test design techniques / ET
All tests in regression set
Code review
85% code coverage

Core Practice # 3: Unit Testing

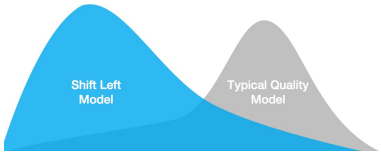


Testing requires independence

Get Developers doing Unit Testing

- Code coverage, Static analysis
- Pair programming, Code reviews
- Test Driven Development ?

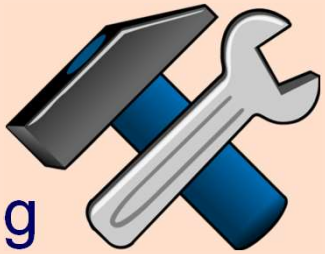
Core Practice # 4: Test Design



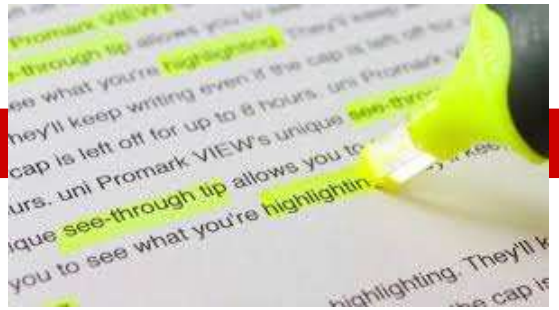
Testing is finding defects

Apply Test Design Identify Test Conditions

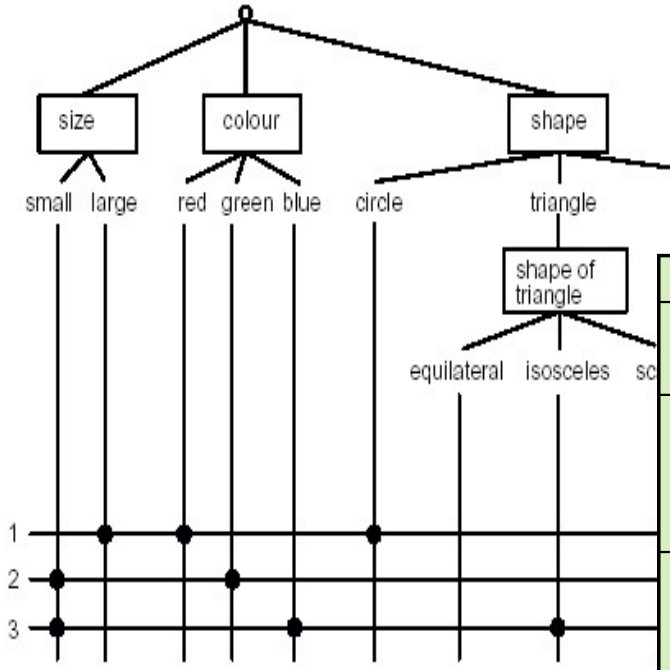
- Test design start-up meeting
- Test techniques as tools
- CTM / Use Cases / Exploratory testing



Apply Test Design



- Identification test conditions
- Establish traceability to requirements



Terms	Valid values	Invalid values
Process status	Idle, Dispatching	Updating
Queue size	Known card type (Visa Card, MasterCard, American Express, Visa Delta)	Empty, unknown card type
Card Number	Numeric, Compliant, Match with card type, valid to date and cardholder name	Empty, not compliant, Mismatch in: Card type, Valid to date and cardholder name.

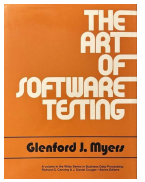
GenAI Test Design

Req. No	Issue Des
01	Business g
01	Not under
01	Value not
02	Subjective other watc
02	Not measu
02	Multiple ideas in one statement
02	Incorrect g
03	No busines
03	No priority
04	Uses "will"
04	No priority
04	No rationa

Attribute	Type	Valid ECs	Invalid EC's
Card type	Set	EC1: Visa Card, Mastercard, American Express, Visa Delta	EC2: Blank (E109), EC3: Other value (E100)
Card Number	Numeric	EC4: Numeric	EC5: Blank, EC6: Non-numeric, EC7 Fails algorithm (E102)
Cardholder Name	Alpha	EC8: Non-blank, alpha ≤ 30 chars	EC9: Blank (E109), EC10: non-alpha (E300): >chars (E300)
Valid To Date	Numeric YY	EC12: Numeric current year ≤ YY ≤ 2030	EC13: Blank (E109), EC14: Non-numeric (E103), EC15: current year, EC16: > 2030 (E104)
Company ID	Alpha	EC17: Non-blank, alpha, registered	EC18: Blank (E109), E19: non-alpha, E20: Unregistered (E200)

TC ID	Objective EC's covered	Card Type	Card Number	Cardholder name	Valid To Date	Company ID	Expected Result
01	Verify all valid inputs (EC1, EC4, EC8, EC12, EC17)	VIsa Card	4111 1111 1111 1111 (valid)	JohnSmith	2025	98775	Processing continues, R02
02	Verify blank card type rejected (EC2)	(blank)	4111 1111 1111 1111	JohnSmith	2025	98775	E109 (blank not allowed)
03	Verify unknown card type rejected (EC3)	ABC Card	4111 1111 1111 1111	JohnSmith	2026	98775	E100 (card type unknown)
04	Verify blank card number rejected (EC5)	Mastercard	(blank)	JohnSmith	2027	98775	E109
05	Verify non-numeric card number rejected (EC6)	American Express	4111 ABCD 1234 1111	JohnSmith	2028	98775	E101 (must be numeric)

Practice # 5: People



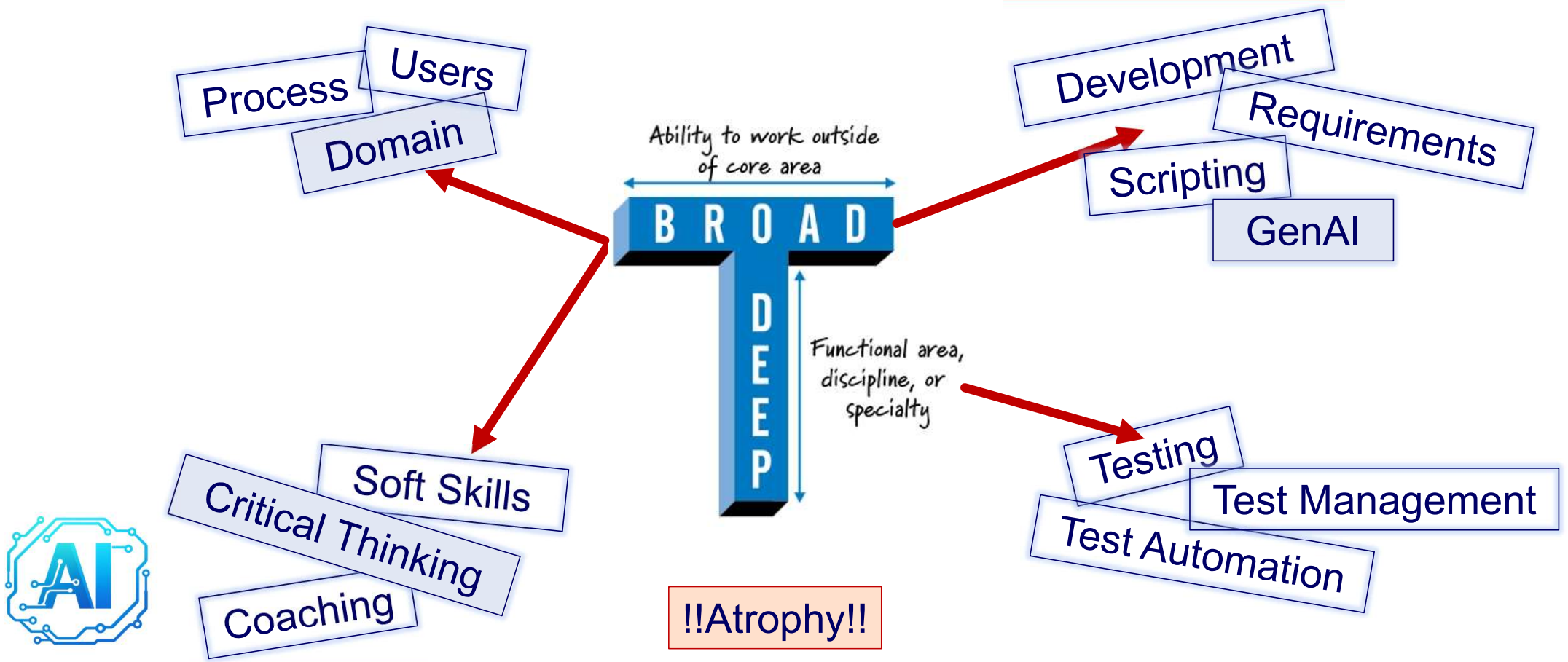
Testing is an extremely complex, intellectually challenging task

Build Experienced and Skilled Testers

- Training (on-the-job), Continuous learning
- Team Dynamics
- Not only for Testers!!

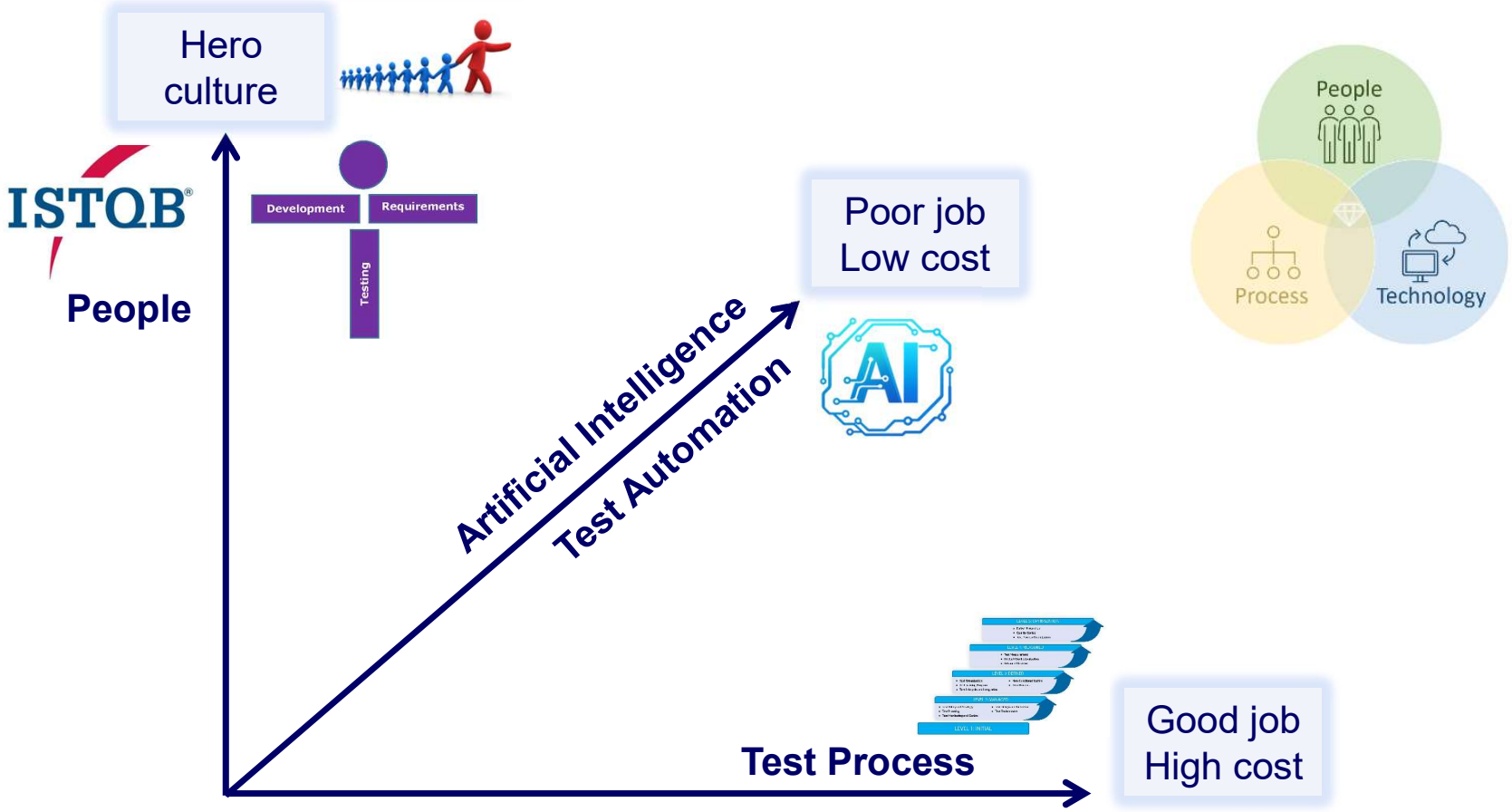
Testers' Skills

Sharing Experiences
Best Practices, Coaching
Workshops and Training

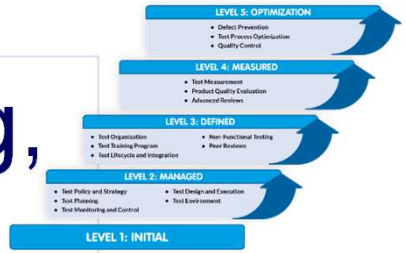


Building on Success

It’s AND …


Hero culture
ISTQB
People
Development
Requirements
Testing
Poor job
Low cost
Artificial Intelligence
Test Automation
AI
Test Process
Good job
High cost
People
Process
Technology

If you don’t know what you are doing, don’t do it with AI.



THANK YOU!

www.erikvanveenendaal.nl

erik@erikvanveenendaal.nl

LinkedIn

erik-van-veenendaal-9663972