High Performance QA

THIRUVENGADAM ASHOK

COUNTERINTUITIVE PERSPECTIVES ALIGNED ON FOUR THEMES OF LANGUAGE, THINKING, STRUCTURE & DOING.

High-Performance QA: Redefining the Path to Clean Code
In today's fast-paced tech landscape, software quality is no longer a luxury–it's a necessity. *High-Performance QA* is your ultimate guide to transforming quality assurance from a routine task into an art form. This book is a blueprint for those seeking to create brilliant code through innovative methods that balance clarity, creativity, and efficiency.

What's inside

✦ **12 Counterintuitive Perspectives**: Dive into fresh, thought-provoking ideas beyond the beaten path of process, technology, and skills.

✦ **Four Transformational Themes**:

(1) **Language**: Master the power of communication to articulate, organize, and express QA strategies with precision.

(2) **Thinking**: Learn to blend logical rigour with creative exploration for smarter problem-solving.

(3) **Structure**: Harness principles of geometry and systematic organization to elevate testing efficiency.

(4) **Doing**: Adopt actionable strategies to achieve more with less effort, including mindfulness-driven testing.

✦ **Mindful QA Methodologies**: Achieve a state of flow where testing becomes intuitive and highly productive.

✦ **Practical Insights and Tools**: From smart checklists inspired by aviation to automation strategies, this book offers actionable techniques that revolutionise testing.

Why this book matters

High-Performance QA isn't just a guide–it's a mindset. Whether you're a developer, tester, or QA manager, this book equips you to:

- Simplify complex challenges.

- Deliver faster and higher-quality results.

- Stay engaged and creative, even under pressure.

- Align QA practices with value-driven outcomes.

Who should read this book?

Perfect for software engineers, team leads, and anyone passionate about optimising software quality, *High-Performance QA* empowers you to rethink what's possible in quality assurance. Transform your approach to QA–embrace clarity, agility, and brilliance with *High-Performance QA*.

About the author

Thiruvengadam Ashok is the CEO of STAG Software Private Limited & Co-Founder of Pivotrics, based in Bengaluru, India. Ashok has dedicated his career to the pursuit of quality assurance in software, continuously evolving his approaches to match the needs of modern systems. He is the creator of HyBIST, an innovative approach to SmartQA that has revolutionised how teams approach hypothesis-driven testing.

Ashok's professional life is deeply intertwined with his personal philosophy. A passionate ultra-distance runner and long-distance cyclist, he applies the principles of endurance and exploration to his work, constantly seeking out new ways to improve software quality. He is also an avid wordsmith, using his love of language to weave both poetry and technical innovation into his life's work.

He holds an M.S. in Computer Science from the Illinois Institute of Technology, a Bachelor's degree in Electronics and Communication Engineering from the College of Engineering, Guindy, and a Postgraduate Diploma in Environmental Law from the National Law School of India University, Bangalore. His life maxim—"Love what you do & Do only what you love"—is reflected in everything he undertakes, from professional projects to personal passions.

Table of contents

Perspective #1 - The power of language

This explores how language enables a mindset of brilliant clarity, essential for high-performance thinking. It highlights how different writing styles, sentence constructs, and sentence types play a key role in the work we do as QA professionals to produce clean code.
—

When we communicate using a language, we use various writing styles, sentence constructs, and sentence types. Are these just tools for writing? Not really. Language isn't only about writing or speaking; it plays a vital role in how we think. A TEDx talk titled "Language Shapes the Way We Think" dives into this fascinating concept. (bit.ly/2LujpZa)

Once we are comfortable with a language, we start to "think" in it. We use different writing styles and sentence types in our minds to understand, describe, specify, evaluate, and report. Language consists of syntax, the rules, and content, the semantics. Syntax shapes both the "what" and "how" of understanding content.

Clarity of Thought: A Cornerstone of High Performance

It requires 'seeing' the system in your mind, running through scenarios that might be of interest, identifying gaps, and posing questions to complete the picture. It involves transforming your mindset to mirror that of the end users and analysing the system through their lens.

Clarity, like water, is clear and transparent. While dirt may colour it, its essence remains unaltered. It flows freely, taking any shape without losing itself. Achieving such Zen-like clarity–where creative juices flow best–relies significantly on language as a vehicle for expressing inner thoughts through various styles.

Let us now explore the language elements, beginning with writing styles and then progressing to sentence types and constructs.

Language Elements: Writing Styles

The four key writing styles are:

(1) Expository: Designed to explain, this subject-oriented style focuses on conveying information about a topic without voicing personal opinions.

(2) Descriptive: Focused on vivid details, this style describes a character, event, or place in great depth.

(3) Persuasive: Aimed at convincing, this style integrates the author's opinions and biases, justifications, and reasoning to persuade others to agree with their point of view.

(4) Narrative: Intended to tell a story, this style often uses characters and recounts events, sometimes from their perspective.

Language Elements: Sentence Types

The four types of sentences are:

(1) Simple or Declarative: States information. Example: "Research suggests that grammar is essential to success."

(2) Command or Imperative: Issues commands or requests. Example: "Open the window."

(3) Question or Interrogative: Poses questions. Example: "Did you complete the assignment?"

(4) Exclamatory: Expresses emotion. Example: "She is going to fall!"

Language Elements: Sentence Constructs

The four forms of sentence constructs are:

(1) Simple Sentence: An independent clause with one punctuation mark at the end. Example: "The essay was late."

(2) Compound Sentence: Joins two related simple sentences. Example: "The essay was late, so he lost marks."

(3) Complex Sentence: Combines an independent clause with a dependent clause. Example: "Because his essay was late, he lost marks."

(4) Compound-Complex Sentence: Combines compound and complex sentences. Example: "Even though the professor was exhausted, she continued grading assignments, and she stayed awake until they were done."

-

Applying Language Elements to QA activities

(1) Understanding: Understanding intent and expectations is essential for effective testing. Adopting the end user's perspective, expository, narrative, or descriptive styles—paired with declarative constructs—clarify user stories. Switching roles to QA, interrogative constructs using expository sentences help fill gaps and refine understanding.

(2) Analysis & Simplification: Simplification reveals true understanding. From a language perspective, it's about summarising complex ideas into concise, short sentences. This sharpens thought and aids focus.

(3) Goal Setting & Planning: Clear goals—'what-to-test' and 'what-to-test-for'—ensure focus and completion. Imperative styles with simple sentences establish clarity. For example, "The system shall provide…" Planning benefits from descriptive styles and imperative constructs for actionable task lists.

(4) Designing: Effective design involves considering conditions and combinations. Imperative constructs, with a choice of sentences based on complexity, suit this task. Evaluation scenarios work best in imperative styles. Example: "Ensure the system does X when Y."

(5) Execution & Reporting: Clear execution instructions in expository styles aid manual or automated scripts. For reporting, declarative constructs describe defects, while imperative constructs outline reproduction steps. Exclamatory constructs are avoided to maintain professionalism.

(6) Analysis & Retrospecting:" Analysis involves summarising insights using declarative and imperative constructs. Persuasive styles are ideal for presenting suggestions.

(7) Note-Taking: This is the freeform part—where observations and past learnings are captured. It combines all styles and constructs, with terseness being key. Non-linear methods like diagrams and mind maps often outperform plain text.

language elements					
language STYLE	Expository, Descriptive, Persuasive, Narrative				
sentence TYPE	Simple, Compound, Complex, Compound-Complex				
sentence CONSTRUCT	Declarative, Imperative, Interrogative, Exclamatory				
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Summary

Clarity reflects the state of mind; great clarity signals an uncluttered mind. The style and structure of sentences shape how we think, understand, design, act, and analyze. Syntax is a trusted guide toward clarity, marking a clear path forward.
In our role as QA professionals producing brilliant code, we navigate challenges such as incomplete information, incorrect data, and time pressures. Language mastery helps cut through these challenges, ensuring we stay on course.

References

- "Four Different Types of Writing Styles: Expository, Descriptive, Persuasive, and Narrative" at https://owlcation.com/humanities/Four-Types-of-Writing

- "Types of Sentences" at http://plato.algonquincollege.com/applications/guideToGrammar/?page_id=3243

- "Sentence Structure" at https://www2.le.ac.uk/offices/ld/resources/study-guides-pdfs/writing-skills-pdfs/sentence-structure-v1.0.pdf

Perspective #2 - Three communication approaches to brilliant clarity

This explores how communication approaches enhance language elements and foster a mindset of brilliant clarity, enabling high-performance QA. Three approaches are outlined here –storytelling (descriptive), rules/criteria (prescriptive), and visual–that play a key role in producing brilliant code from a QA perspective.

The background

Testing is fascinating because it's unbounded. Customer needs and expectations often remain incomplete, evolve constantly, and shift rapidly alongside technology. A nimble process model backed by great technology helps–but only when the problem is well-analysed, a solution effectively synthesised, and the implementation executed flawlessly.

What enables us to analyze a problem, craft a great solution, and implement it effectively?

Great clarity stems from effective communication.

In the previous chapter, I discussed how sentence types, constructs, and styles aid problem-solving for brilliant code. Specific language combinations help activities succeed. Here, I'll dive into another dimension–communication approaches–that builds upon styles, types, and constructs. These approaches include storytelling, rules, and visuals. Choosing the right approach depends on the problem-solving phase–analysis, synthesis, or implementation. Let's explore them in detail.

language elements					
communication APPROACH descriptive, prescriptive, visual					
language STYLE Expository, Descriptive, Persuasive, Narrative					
sentence CONSTRUCT Declarative, Imperative, Interrogative, Exclamatory			sentence TYPE Simple, Compound, Complex, Compound-Complex		
problem ANALYSIS		solution SYNTHESIS		implementation MANAGEMENT	
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Problem Analysis Phase: Descriptive Approach

The key to effective problem analysis is understanding the problem deeply. But how do we achieve that?

Think back to childhood stories—they helped us grasp concepts like good versus evil or right versus wrong. Storytelling connects elements in an engaging, human way, enabling better understanding and sparking meaningful questions. This connection aids deeper understanding, which I call the **descriptive approach**.

1. **Understanding**
   We grasp systems by reading specifications or exploring prior versions. Writing user stories is a form of storytelling, describing what a system should do. While specifications can feel dry, adding a human touch—who uses it, why they need it, what they value, when and how they use it—makes them relatable. Storytelling helps describe systems, leading to better questions and insights.

2. **Analysis and Simplification**
   Storytelling simplifies problem analysis by identifying personas, testing elements, key attributes, and environments to consider. It also clarifies whether we're dealing with new or modified systems.

language elements					
communication APPROACH descriptive, prescriptive, visual					
language STYLE Expository, Descriptive, Persuasive, Narrative					
sentence CONSTRUCT Declarative, Imperative, Interrogative, Exclamatory			**sentence TYPE** Simple, Compound, Complex, Compound-Complex		
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Solution Synthesis Phase: Prescriptive Approach

Synthesising solutions involves identifying and combining conditions to create clear, actionable rules–a prescription. This **prescriptive approach** helps us extract conditions and formulate solutions.

1. **Goal setting & planning**
   This phase involves defining clear, precise acceptance criteria.

2. **Designing**
   Understanding system behavior often starts with a descriptive dive into testing entities. Once conditions are identified, test scenarios are designed by combining these conditions, forming a prescriptive blueprint for evaluation. This applies to both functional and non-functional scenarios.

language elements					
communication APPROACH descriptive, prescriptive, visual					
language STYLE Expository, Descriptive, Persuasive, Narrative					
sentence CONSTRUCT Declarative, Imperative, Interrogative, Exclamatory			**sentence TYPE** Simple, Compound, Complex, Compound-Complex		
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Implementation Management Phase: Visual Approach

Execution generates vast data, making visuals essential for effective reporting and analysis. Structured graphs, combined with unstructured visuals, enhance clarity and break monotony. The **visual approach** is further enriched by storytelling, which provides context to data, and prescriptive elements, which define limits and standards.

1. **Execution and Reporting**
   Visual aids like charts simplify progress tracking and risk assessment.

2. **Analysis and Retrospecting**
   Adding storytelling to visuals expands understanding, while prescriptive clarity ensures actionable insights, fostering superior management.

language elements					
communication APPROACH descriptive, prescriptive, visual					
language STYLE Expository, Descriptive, Persuasive, Narrative					
sentence CONSTRUCT Declarative, Imperative, Interrogative, Exclamatory			**sentence TYPE** Simple, Compound, Complex, Compound-Complex		
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Summary

Clarity reflects an uncluttered mind. Sentence style and structure influence how we think, understand, design, act, and analyze. Communication—how we string sentences—determines our ability to see the full picture and identify gaps.

A thoughtful blend of human-friendly storytelling, structured prescriptive approaches, and fact-rich visuals broadens cognitive capabilities. It lets us see the big picture and zoom into details when needed.

Perspective #3 - To express well, choose the right medium

Earlier, I examined how communication approaches, language styles, and sentence constructs play a key role in shaping the activities we do as producers of brilliant code from a QA perspective. Here, I explore the role of the medium in expressing thoughts and ideas effectively, enabling action. A loose, frictionless medium like paper is ideal for early-stage ideas, while a strict template or tool suits fully developed, clear ideas.

So, what is the role of the medium?

Earlier, I focused on 'communication approach,' outlining three approaches—descriptive, prescriptive, and visual. Each is better suited to specific phases of problem-solving. Now, let's remember that thoughts and ideas germinate in our minds and are expressed through a medium, like a sheet of paper or a tool/template. The key distinction lies in structure: some media are unstructured, while others are rigidly structured.

If there's a mismatch—or 'impedance'—between thought and medium, expressive power is greatly diminished. In other words, medium matters.

What are the various media of expression?

When expressing something in its early stages of formulation, it's best to do this quickly and refine continuously. An unstructured medium, like pen and paper, lets us stay creative by removing boundaries—allowing free arrangement of text, the use of pictures, or even scattered thoughts on a page.

When the thought has become clear and well-formed, it's better to express it in its entirety without losing any details. This is best achieved using a structured medium, like a template or a digital tool.

language elements					
medium of EXPRESSION paper(unstructured), tool/template(structured)					
communication APPROACH descriptive, prescriptive, visual					
language STYLE Expository, Descriptive, Persuasive, Narrative					
sentence CONSTRUCT Declarative, Imperative, Interrogative, Exclamatory			**sentence TYPE** Simple, Compound, Complex, Compound-Complex		
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplication	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

How do these connect?

"Ideas germinate and bubble, enabled by language style. They connect to form thoughts that coalesce and, aided by the communication approach, turn into reality—finally expressed through the medium."
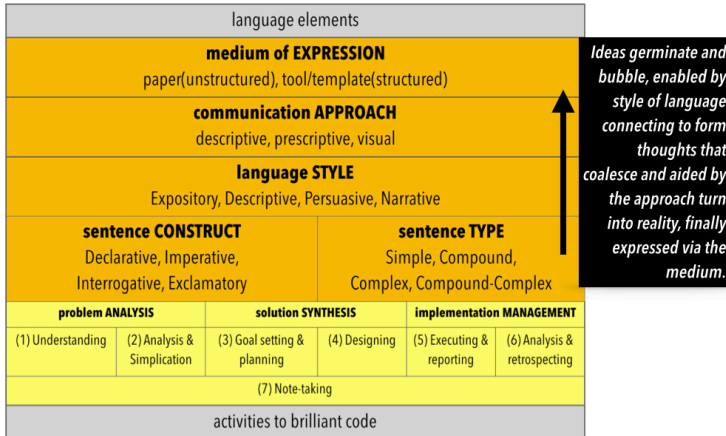
What medium suits each communication approach?
Descriptive Communication

For early-stage thoughts or ideas, a descriptive style thrives with unstructured media. Paper is a natural choice—free-flowing and flexible. You can write sentences in any direction, connect them with pictures, use colours liberally, and even create mind maps or sketches. This taps into the creative right brain, refining the problem and clarifying the idea.

Examples:

- Mind maps of system landscapes, connecting users (personas), system entities (features, flows), environments, and attributes.

- Jottings of ideas, potential issues, questions, clarifications, or flow diagrams.
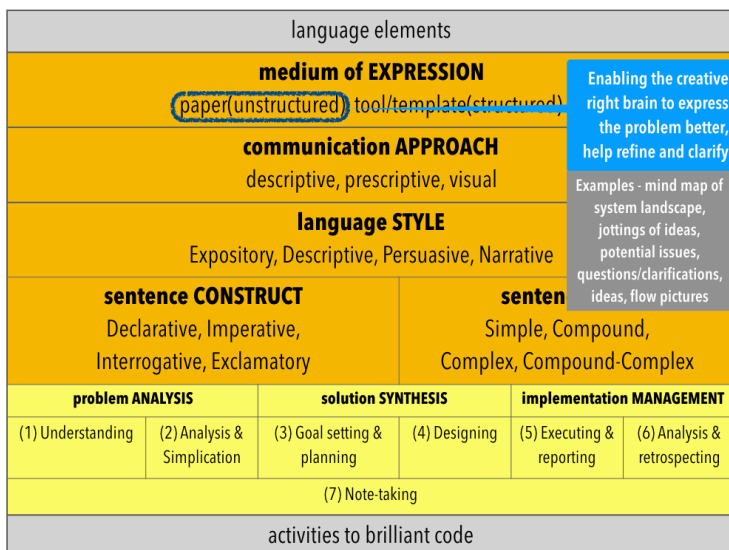
Prescriptive Communication

Once the solution is reasonably clear, it's time to focus on capturing the details fully and clearly. A **structured medium** is the best choice here—a well-formed template on paper or a digital tool ensures nothing is missed, and clarity is preserved in its entirety.

Examples:

- Spreadsheets or specialised tools for outlining scenarios.
- Templates for bug reports or structured documentation.

Visual Communication

The choice of medium in visual communication depends on the maturity of the thought.

- If ideating, paper is often the first choice, offering flexibility.
- If representing rich, factual data, a tool is more suitable for precision and clarity.

Summary

The medium we choose to express our thoughts—driven by our language and communication approach—is crucial to effective expression. Clear, well-structured communication ensures clarity for ourselves and others.

Any friction that impedes free-flowing thoughts is unwelcome. That's why the medium of expression plays a vital role in transforming thoughts into actionable ideas. Loose, frictionless media like paper are best for early-stage ideation, while structured tools or templates excel at capturing fully-formed, detailed concepts.

Perspective #4 - Left brain thinking to building clean code

This perspective, under the 'THINKING' theme, examines how logical 'left-brained' thinking plays a key role in producing brilliant code, especially from the QA angle.
—

The Challenge of Intelligent Testing

Testing is about intelligently perturbing a system to shake out issues that may exist. The challenge lies in knowing if all the issues have been uncovered. A logical approach to identify both good and erroneous scenarios is essential to justify the completeness of validation. At the same time, creativity and contextual understanding are necessary to perturb the system effectively. Injecting a dose of randomness into this process serves as the final push toward thoroughness.

Logical Thinking in Testing

In this article, we delve into how logical 'left-brain' thinking applies to testing a system. We break down validation as a problem-solving approach comprising SEVEN activities.

Let us explore the role of logical thinking in:

- System understanding (1 & 2)

- The act of validation (3 & 4)

- The interpretation of results (5 & 6)

- Note taking (7)

problem ANALYSIS		solution SYNTHESIS		implementation MANAGEMENT	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					

How do we think logically? Let's consider three key aspects:

- The style of thinking

- The methods we use

- The approach we base it on

"thinking" approach					
logical "left brain" thinking					
thinking STYLE Forward, Backward, Approximately		**using METHODS** techniques, principles		**based on APPROACH** process, habits, experience	
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

We can analyze a problem by:

- Breaking it down logically in a forward manner
- Setting the goal and working backward
- Approximating and refining as we go

"Forward" Thinking Style

The 'forward' thinking style involves breaking down the system logically to define 'what-to-test.' This means systematically decomposing the system into key business flows handled by various personas, then aligning these with business requirements, which are essentially collections of technical features.

"Backward" Thinking Style

The 'backward' thinking style is useful for setting expectations, defining criteria, and establishing goals for uncovering potential issues.

Take criteria as an example. How do we ascertain the purity of a material? First, we define the expected purity by identifying the properties it should satisfy. Then, we check if the material meets those properties. Similarly, in software, properties align with quality models like ISO 25010, which outline various '-ities' needed to meet quality requirements.

What is the goal of clean code? It is to ensure no issues disrupt the end user's experience. Hypothesising potential issues starts with understanding the system's intended behavior, its environment, interactions with other systems, the technology used, and the inputs it processes. This allows us to set a goal, refine it, and work toward it systematically.

"Approximation" Thinking Style

When some details are unknown and need discovery, approximation proves useful. This is particularly valuable when estimating data volumes, transaction loads, and similar metrics. One of my favourite questions in workshops is, "How many hairs do you have on your head?" I'm always amazed at the answers, some say 10,000 while some enthusiastic participants say ONE million! I nudge participants to approximate scientifically—using simple calculations like area multiplied by hair density—and then refine their estimates.

Role of Methods & Approach

In forward, backward, or approximate thinking styles, methods and approach play a critical role.

Applying Techniques

In forward, backward, or approximate thinking styles, methods and approach play a critical role. Applying well-formed techniques, particularly in test design (4), is well-established. These techniques, broadly categorised as black-box or white-box, rely on principles that enhance testability by clarifying specifications and criteria.

A Disciplined Approach

Ultimately, the approach—rooted in a disciplined thought process and good habits—is key to leveraging logical left-brain thinking. Systematic retrospectives, learning from past experiences, and applying these insights to current challenges are hallmarks of mature thinking.

"thinking" approach					
logical "left brain" thinking					
thinking STYLE Forward, Backward, Approximately		**using METHODS** techniques, principles		**based on APPROACH** process, habits, experience	
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Summary

Logical 'left-brain' thinking is essential to effective testing. Testing is not just an act but an intellectual examination of what might go wrong and how to perturb a system effectively and efficiently. This approach blends forward, backward, and approximate thinking styles, supported by well-formed techniques and high-level principles. It is grounded in disciplined processes, strong habits, and the ability to learn and adapt from experience.

Perspective #5 - It takes right-brain thinking to go beyond the left

In the second perspective on thinking, I explore how creative, right-brain thinking helps us transcend logical thought to uncover new paths or discard ineffective ones, ultimately improving outcomes and delivering brilliant code.

—

The Limits of Logical Thinking

Logical, structured, and organised thinking is invaluable. It gives us clarity and confidence in what needs to be done. It allows us to plan and execute with purpose. A left-brain approach lends credibility to test design and provides a measurable sense of coverage. But is it enough?

Our activities may feel comprehensive, but will the outcomes match?

Relying on specifications, techniques, and processes is scientific and takes us far. Yet testing is about uncovering the unknown. So how far is "good enough"? Testing involves intelligently perturbing a system to uncover hidden issues. This requires navigating many paths through the application—far too many to plan for comprehensively.

Where Creativity Comes In

This is where right-brain creative thinking becomes crucial, enabling us to go beyond the logical. It helps us vary paths, discover new ones, and improve outcomes. This isn't about randomness or ad-hoc approaches, though randomness can sometimes help. It's about starting with logical, organised thinking, infusing it with creativity, and occasionally embracing unexpected detours.

Start with logical thinking, get creative and finish off with Ad-hoc thinking

Testing is a curious discipline—it demands clairvoyance to foresee the unknown, the ability to perceive what's missing, and the rigour to comprehensively assess what's present, ensuring nothing slips through the cracks.

What does it take to think creatively? Creative thinking can be approached in three ways: visual thinking, contextual thinking, and social thinking.

The Three Pillars of Creative Thinking

Creative thinking can be approached in three ways:

1. Visual Thinking
2. Contextual Thinking
3. Social Thinking

"thinking" approach					
			creative "right brain" thinking		
visual thinking colours, shapes, spatial		**contextual thinking** observe, experiment, react		**social thinking** feeling, empathy	
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Visual Thinking: See the Problem and the Solution

Visual thinking is a key trait of engaged thinkers, allowing one to see problems, solutions, and gaps more clearly. It is about approaching challenges spatially and non-linearly, often aided by tools like mind maps, doodles, and sketches. These visuals, enhanced with colours and styles, stimulate the right brain, helping us uncover new connections.

This approach sharpens our understanding, allowing us to see the big picture without getting lost in the details. It creates a tangible representation of ideas that can be analysed, questioned, and refined. Visual thinking is especially useful for envisioning potential behaviours and crafting scenarios beyond those born of purely logical thinking.

It also proves valuable in later stages of reporting, enabling better analysis and improvement of activities.

"thinking" approach					
				creative "right brain" thinking	
visual thinking colours, shapes, spatial		**contextual thinking** observe, experiment, react		**social thinking** feeling, empathy	
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Contextual Thinking: Adapt to the Situation

Disciplined, left-brain thinking can sometimes trap us in predictable patterns. Testing, however, requires adaptability, variation, and sensitivity to context. Being attuned to the environment heightens observation, encourages experimentation, and helps us adjust rapidly.

This form of thinking grounds us in the present context, deepening understanding and prompting sharper questions. It's particularly useful during stages of understanding and analysis (1-2), test design (3-4), and efficient validation (5-6).

"thinking" approach					
				creative "right brain" thinking	
visual thinking colours, shapes, spatial		**contextual thinking** observe, experiment, react		**social thinking** feeling, empathy	
problem ANALYSIS		**solution SYNTHESIS**		**implementation MANAGEMENT**	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Social Thinking: Empathise with the End User

Ultimately, applications are consumed by real people, either directly or indirectly. Whether humans use the application directly or through a system, the end user's experience is paramount.

Social thinking shifts us into a "feeling mode," where we empathise with end users. This perspective is critical not just in test design and evaluation but also in planning how much testing to do, assessing risk, and prioritising issues for resolution. Empathy is a cornerstone skill for testers, allowing us to step into the shoes of real users.

"thinking" approach					
			creative "right brain" thinking		
visual thinking colours, shapes, spatial		contextual thinking observe, experiment, react		social thinking feeling, empathy	
problem ANALYSIS		solution SYNTHESIS		implementation MANAGEMENT	
(1) Understanding	(2) Analysis & Simplification	(3) Goal setting & planning	(4) Designing	(5) Executing & reporting	(6) Analysis & retrospecting
(7) Note-taking					
activities to brilliant code					

Summary: The Balance Between Logic and Creativity

Logical, left-brain thinking is essential for effective testing. But right-brain creativity is what helps us go further–varying paths, discovering new ones, and enhancing outcomes. Creative thinking involves:

- **Visual thinking:** Using visuals to think spatially and explore connections.
- **Contextual thinking:** Reacting, experimenting, and questioning based on context.
- **Social thinking:** Empathising with users to prioritise and evaluate effectively.

By blending logic and creativity, we elevate testing from competent to brilliant.

Perspective #6 - The ultimate power lies in the "empty" middle

The final perspective on the theme of THINKING explores the infinite power we possess, which resides neither in left- nor right-brained thinking but in the middle—in the pure silence of mindfulness.

—

The Focus on Outcomes

Activities are essential, but outcomes are what matter today. Delivering value is paramount. Increasing expectations from customers and the drive to achieve more with less have made outcomes increasingly challenging. In our fast-paced world, where software is released continuously, tools are central to rapid validation. Yet, the hunger for speed is relentless, and there can be no compromise on the outcome—quality.

Balancing Logic and Creativity

Logical and creative thinking play a crucial role in ensuring the activities we undertake are effective and efficient, driving good outcomes. But is it enough to simply think logically, be creative, and leverage tools? Outcomes might be excellent, but do they truly fulfil the promise to the customer? It's about keen observations and subtle adjustments to ensure outcomes deliver greater value. It's about being in the *flow*.

The Role of Repetition and Automation

Testing involves numerous activities, some requiring logical and creative thinking, others repetitive tasks that demand diligence. Left- and right-brain thinking support the former, while the latter benefits from structured processes and the appropriate use of automation.

The monotony of processes can make them tedious, often leading to slippage. We might perceive this as stifling creativity. But does being "mindless" make one dull or uncreative? Let's look at it differently.

Discovering Flow Through Mindfulness

Being "mindless" requires you to empty your mind so you can be present, attaining a state of mindfulness—a state of flow. I discovered this in endurance cycling. Endurance cycling involves covering long distances, often hundreds of kilometres in one stretch. Like any endurance activity, it's less about physical strength and more about mental resilience.

When cycling for hours non-stop, fixating on the goal or the distance left can make the activity monotonous, mentally draining, and physically exhausting. It robs the joy of cycling. I found that by focusing solely on the wheel in front of me and the rhythm of my pedal rotations, rather than the mile markers, I was entirely in the present. No attachment to the past, no anxiety about the future. Just blissful presence—effortless and peaceful.

The mind emptied. Time stopped. I was in the *flow*. A state of awareness anchored in the now. Sheer joy. This is where peak potential resides–when you're so attuned and observant that you respond to your environment with remarkable agility, seemingly without effort.

Mindfulness Enhances Testing Outcomes

Staying in the middle through mindfulness complements left- and right-brain thinking, significantly enhancing outcomes. The effortless observation of the now allows you to immerse fully in the task, absorbing every nuance, processing it, and responding in a way that feels almost magical–devoid of resistance or strain.

HyBIST: The Practice of Mindful Testing

Hypothesis Based Immersive Session Testing (HyBIST) builds on this concept. It extends the logical and scientific foundation of Hypothesis-Based Testing into short, immersive sessions of 60–90 minutes with a special focus on mindfulness. It's about being light and nimble, using concise language, fostering curiosity, questioning assumptions, and continually re-establishing connections. This approach blends logical and creative thinking with lightweight tools for a sharp, focused experience.

The Equation for Productivity

A mindful session of testing is akin to this equation:

$$P = Lt_{(t \to 0)} W/t,$$

where P = Productivity, W = Work to be done, and t = Time.

When time feels frozen, productivity soars. This means staying acutely conscious of the now, focused on the present, and fully enjoying the testing process. Mindfulness transforms testing from a chore into a fulfilling, enjoyable act.

Unlocking Infinite Power in the Present

While left-brain logic, diligent processes, tool exploitation, and right-brain creativity are vital, they remain limited. The infinite power we all have lies in the "middle"–in freezing time and discovering the silence of the present.

A Simple Experiment in Mindfulness

Want to experiment with mindfulness? Take a glass of water and sip it slowly, focusing on the act. Feel the water touch your lips, swirl on your tongue, and flow down your throat. Spend a few minutes in this experience, and you'll enjoy it. Time stops. Wouldn't it be amazing to approach testing sessions this way?

A Final Thought: Magic is in the Middle

For a great future, stay in the present. To move swiftly and achieve more, sometimes you need to pause. *The magic lies in the middle.*

Summary- Harness the Power of Presence for Productivity

Achieving more with less is the need of the hour. Test automation has emerged as a key enabler for speed, freeing up time for greater pursuits.

Thinking smarter with left-brain logic, creatively with right-brain flair, and leveraging tools for efficiency is valuable, but true brilliance happens when you're fully immersed in the task. Being mindful allows you to think clearly, respond swiftly, and create freely without being shackled by the past or anxious about the future. Ultimate power lies in the middle—in the now.

Perspective #7 - The Power of Geometry

This perspective, the first in the theme of STRUCTURE, explores how the organisation of elements plays a key role in achieving more with less. In the subsequent two perspectives, we'll delve deeper into the arrangement of product elements and test artefacts, examining how they aid in clear thinking and deliver high performance.
–

Geometry: The Foundation of Structure

Geometry is a branch of mathematics concerned with questions about shape, size, the relative position of figures, and the properties of space. In static structures, the shape is crucial for strength and aesthetic appeal. In dynamics, geometry enables higher power transfer and output with lower energy expenditure.

Running Form: Efficiency Through Shape

Let me share a story about running to illustrate this. When you venture into long-distance running–marathons or ultramarathons–'running form' becomes critical. So, what is 'running form'? It is the erect, lean-forward stance of a runner combined with clean forward-backward hand movements.

How does this help? The erect, lean-forward stance leverages gravity to propel you forward, while rhythmic hand movement engages the core, reducing the burden on your legs. When I adopted this running form, my performance improved significantly.

But what does it take to master the form? First, understanding the science behind it. Then, learning the techniques to achieve it, followed by relentless practice until it becomes second nature. The result? Faster, longer runs with reduced injury risk and quicker recovery–a striking improvement simply by adopting the correct form.

Frame Geometry: The Cyclist's Edge

As an endurance cyclist, I once sought to improve my average speed on long rides. Over the years, I had honed my stamina, nutrition, strength, and mindfulness to cover longer distances. However, I wanted to go faster.

The solution? Switching from my relaxed hybrid bike to a race geometry bike with drop-bar handlebars and a sleek frame. My first ride was a revelation—a 4-5 km/h improvement in average speed. Nothing about me had changed; the difference lay in the bike's geometry.

The new frame shifted my stance, enabling better power transfer to the pedals with the same energy input. The initial rides were challenging—my back protested against the aggressive posture. But as my muscles adapted, the rides became smoother and faster. Once again, knowledge of the technique and consistent practice made all the difference.

Geometry in Testing

Now let's apply these ideas to software testing. In both running and cycling, the shape or form directly influenced efficiency and performance. Similarly, geometry in testing involves the arrangement of elements for better outcomes.

Arrangement of SUT Elements

The geometry of the System Under Test (SUT) refers to viewing it as a set of flows used by different end users. Each flow comprises business requirements, which are delivered by features built upon structural components.

This 'geometry' allows for clean problem decomposition, enabling us to clearly understand end-user expectations and how the system is constructed. This structure helps uncover what we know and, more importantly, what we don't–unaddressed gaps that could lead to future defects.

Arrangement of Test Cases

Similarly, the geometry of test cases involves grouping them to target specific types of defects. Visualise this as a multi-layer filter, where each layer addresses a particular defect type.

This structured arrangement simplifies test design, making it sharper and more comprehensive. The geometry of the SUT sparks intelligent questions to uncover potential bugs, while the geometry of test cases enhances defect filtration capabilities.

The next perspective explores these arrangements in greater detail.

Geometry in Nature: Beauty and Power

Nature employs geometry with remarkable efficiency and elegance. Shapes provide strength, utility, and aesthetic appeal. From the honeycomb to cathedral domes, roller coasters to DNA's double helix, geometry is nature's tool for achieving harmony between form and function.

In testing, shaping the problem well allows for deeper understanding and comprehensive evaluation. By questioning deeply, understanding thoroughly, and structuring intelligently, we can harness the power of geometry for higher yield and efficiency.

Summary

A good running form and an optimised cycling geometry demonstrate how higher performance can be achieved without additional power output. Similarly, in QA, it's not just the content of test artefacts—like scenarios, cases, or plans—that matters. It's how they are structured and organised that enables us to accomplish more with the same or fewer resources.
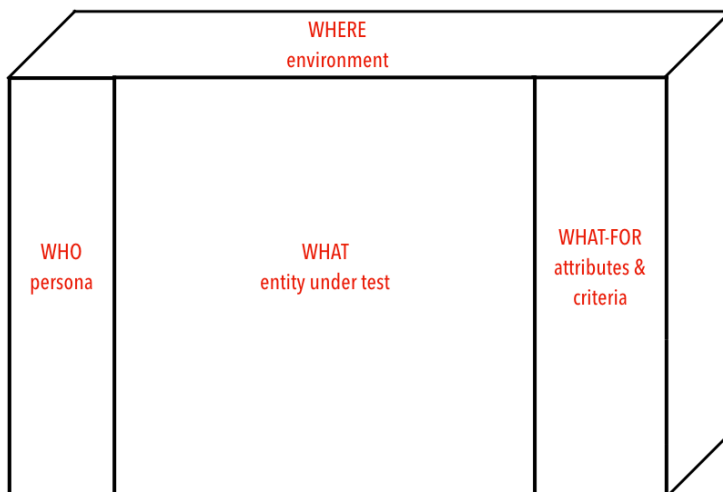
Perspective #8 - The 4Ws to Structuring a Problem Well

In this perspective, I explore how arranging the elements of a system-under-test enables us to clearly see the problem in terms of facts and questions, setting the stage for efficient evaluation.

Structuring a Problem: The Key to Effective Solutions

In problem-solving, articulating the problem well—or structuring it effectively—is crucial for arriving at a great solution. In the QA context, what are we trying to achieve? Simply put, we aim to identify the end users and their needs, understand their expectations for fulfilling those needs, and determine the environments where the system should function.

A good problem statement in QA outlines four key aspects:

1. **WHO** are the end users (personas)?
2. **WHAT** entities need to be tested?
3. **WHAT-FOR** (attributes and criteria) do these entities need to be tested?
4. **WHERE** (environments) will these tests take place?

This approach can be distilled into a straightforward framework: the **4Ws—WHO, WHAT, WHAT-FOR, WHERE**. Let's examine each component in detail.
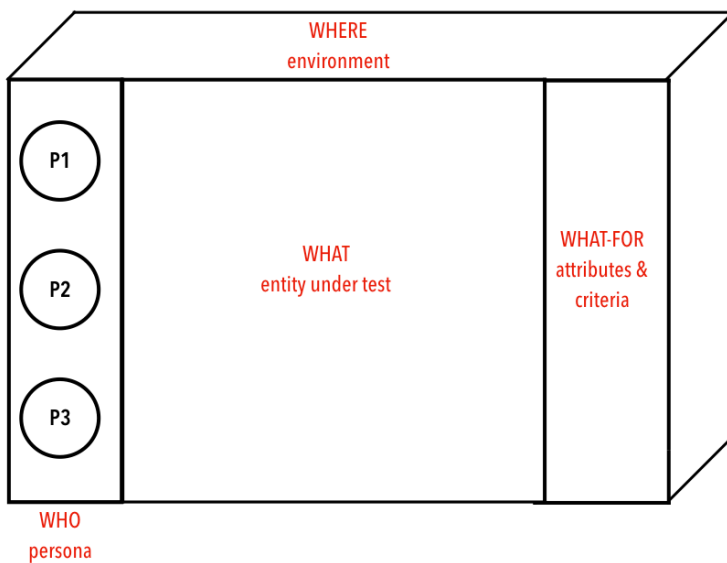

The WHO

Who are the consumers of the system? These are the end users, or personas, who will interact with it. Keep in mind, the end user may not always be a human—it could also be a machine. Starting with the WHO helps us understand the intended audience and tailor our approach accordingly.

The WHAT

This aspect refers to the entities that make up the system. These entities can exist at
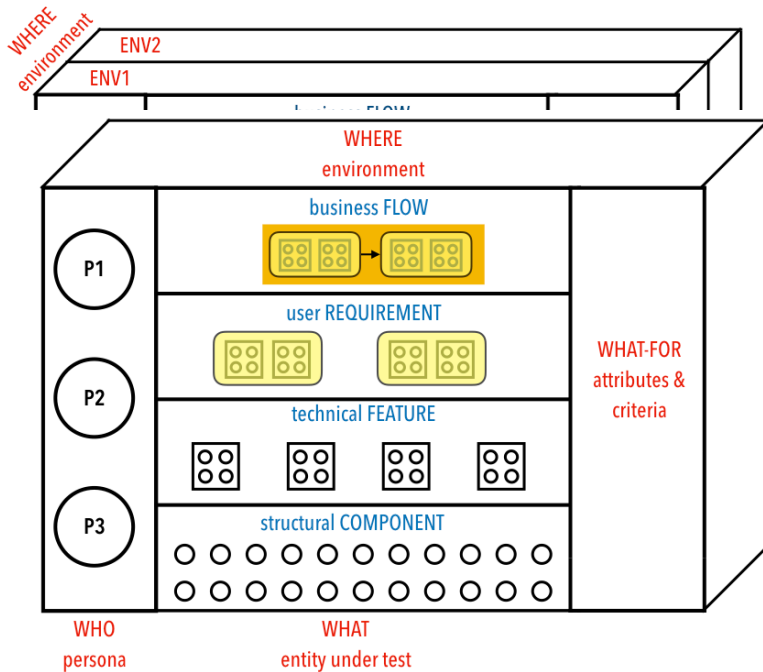
different levels of granularity:

- Structural components (building blocks typically termed as units).
- Technical features, representing basic system behaviours.
- User requirements or use cases, tied to specific personas.
- Business flows, which are sequences of user operations at a higher level.

Clearly identifying the WHAT allows us to focus on the relevant parts of the system.

The WHAT-FOR

Once we know WHO and WHAT, we need to establish what these entities are expected to satisfy—their purpose. This involves defining attributes such as functionality, performance, and security. What's crucial is attaching measurable criteria to each attribute, ensuring they are testable and meaningful.

The WHERE

Finally, we address WHERE the system will operate. This includes environments like operating systems, browsers, and devices. Variations in environments might lead to different end users, attributes, criteria, and even test entities. Recognising these distinctions is vital for comprehensive evaluation.
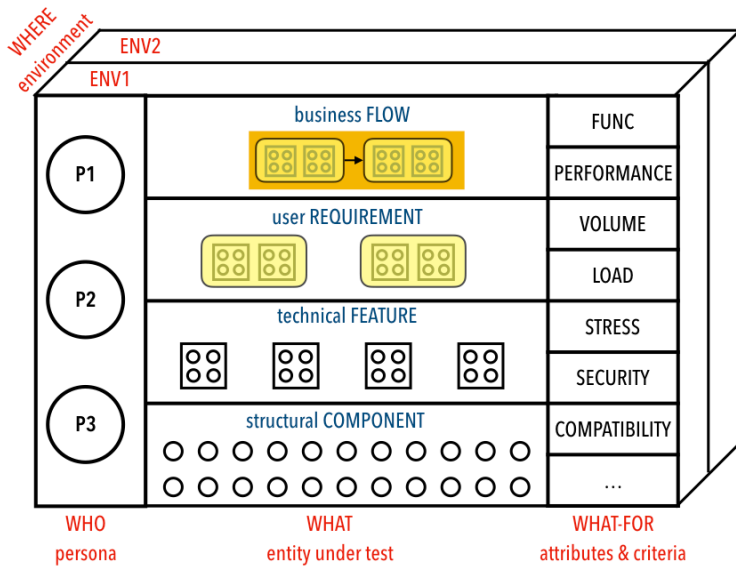
The Power of a Well-Structured Problem

A thoughtful decomposition and structuring of the problem are key to achieving high performance. This approach brings clarity, enabling a deeper understanding of the problem and paving the way for an efficient solution.

All this takes is "4Ws : WHO | WHAT | WHAT-FOR | WHERE".

Summary

High-performance QA begins with decomposing the problem effectively. This means simplifying the evaluation process into a clear framework:
Test WHAT granularity of entities, to meet WHAT-FOR attributes and criteria, for WHOM, and in WHAT environments.

Or, in simpler terms: the 4Ws–WHO, WHAT, WHAT-FOR, WHERE.

Perspective #9 - Ten Ways to Smartly Organise Test Assets

This examines how organising test assets–test scenarios and cases–plays a crucial role in enhancing the effectiveness and efficiency of testing.

The Roller Coaster Analogy

Riding a roller coaster is an exhilarating experience. Regardless of age, the thrill of its twists and turns is unmatched. The slow climb, followed by a sharp drop and gut-wrenching twists, fills us with awe and respect for this engineering marvel.

'Six Flags Magic Mountain - Pic courtesy Beyond Neon/Flickr/CC BY 2.0'

How does this steel lattice deliver such thrills while maintaining a safety record better than a toaster? The answer lies in its **form and structure**. The integration of design and engineering ensures the ride is both thrilling and safe.

"Form and Structure" are omnipresent, from nature's leaves and honeycombs to man-made structures. Form is about external shape and presentation, while structure is about the arrangement and composition of elements. Both are essential for any design activity, including test design.

Beyond the Content of Test Cases

We often associate effective testing with good test cases, typically focusing on their content. But is the "goodness" of a test case purely a result of experience or robust techniques?

The architecture–comprising the external **form** and internal **structure**–is key to ensuring test cases are adequate, optimal, and rapidly generated or automated. Form shapes test cases, enabling a view of adequacy, review-ability, automate-ability, and product health. Structure, on the other hand, combines elements into a coherent whole, allowing us to explore these aspects effectively.

The Three Key Aspects of Organisation

Smart organisation of test assets revolves around three critical aspects:

1. Product Aspect
2. Test Aspect
3. Business Aspect

organise by **PRODUCT ASPECT**	organise by **TEST ASPECT**	organise by **BUSINESS ASPECT**
1 test WHAT by ENTITY	**4** evoke WHAT BEHAVIOUR, excite with WHAT STIMULI by GRANULARITY	**7** whose POV tagged to PERSONA
2 satisfy WHAT by LEVEL	**5** for WHAT INTENT - conform or break by INTENT	**8** for WHAT STATE grouped into SUITES
		9 how IMPORTANT ordered by PRIORITY
3 test FOR-WHAT by TYPE	**6** satisfy CRITERIA or find ISSUE by OBJECTIVE acceptance CRITERIA potential ISSUE	**10** test WHERE mapped to ENVIRONMENT

Organisation by Product Aspect

This involves structuring test scenarios and cases at the right granularity, whether they focus on small components or large business flows. Organisation is based on:

- **Entities under test:** Define what to test.
- **Quality levels:** Clarify test-for-what.
- **Test types:** Determine when to test.

These principles, derived from Hypothesis Based Immersive Session Testing (HyBIST), bring clarity to the testing process.
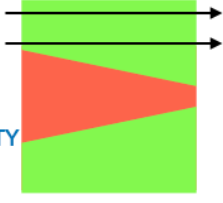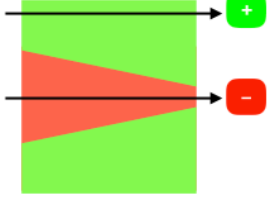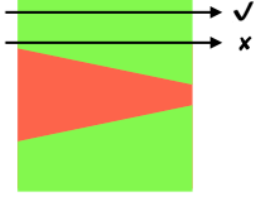
Organisation by Test Aspect

This aspect disambiguates behaviour and stimuli. Here's how:

- **Scenarios (behaviour)**: Conditions or flows being tested.
- **Cases (stimuli)**: Data sets to stimulate behaviour.

The process involves:

1. Ensuring the right granularity for scenarios and cases.
2. Grouping assets based on intent: conformance-oriented (positive) or robustness-oriented (negative).
3. Associating each scenario with its purpose–criteria to meet or issues to uncover.

organise by **PRODUCT ASPECT**
organise by **TEST ASPECT**

4 by GRANULARITY test SCENARIO test CASE		Generate scenarios by combining the behaviour conditions optimally and then the test cases by the combining the input stimuli optimally for each scenario. A scenario tests a specific behaviour flow with test cases ensuring the full flow is well covered. Group behaviours and then the stimuli for each behaviour
5 by INTENT CONFORMANCE ROBUSTNESS	+ −	Mark test scenarios and cases as to whether they check for conformance to intended behavior (positive) or assess robustness in erroneous situations. This ensures that test scenarios and cases are indeed defect seeking and not merely conformance oriented. Be defect seeking in addition to conformance orientation
6 by OBJECTIVE	✓ ✗	Associate test scenarios and cases with what (acceptance) criteria they are attempting to evaluate or to potential issues that they are seeking to uncover. This allows one to be objective focused as to meet or break. Be purposeful as to what issues to uncover or criteria to satisfy.

3. Organisation by Business Aspect

This is about adopting a user-centric view:

- **Tagging scenarios to personas**: Evaluate from the end user's perspective.
- **Group scenarios into suites**: Include scenarios across entities, levels, and types.
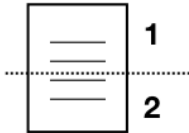- **Prioritising scenarios**: Enable optimal execution and automation.

The Complete Picture of Organisation

Organising test assets using these three aspects–Product, Test, and Business–offers a comprehensive ten-way view across the lifecycle. This structured approach sharpens clarity, addressing questions like:

- Is it adequate?
- Is it optimal?
- How can we evaluate just enough?

This holistic view, especially during design and evaluation, empowers you to design and organise test assets intelligently.

organise by **BUSINESS ASPECT**

|		
7 tagged to **PERSONA**		Tag the test scenarios and cases to the various persona so as move to a human centred evaluation from product centred evaluation. This allows us to ultimately ensure expectations and not just needs are met. Test to meet EXPECTATIONS and DELIVER VALUE
8 grouped into **SUITES**		Group similar test scenarios and cases into a suite that may be automated. The suites could be basic health checks, sanity suites and may cut across levels, types and entities. Suites enable us to evaluate of the system has reached a 'certain state'. i.e. basic health OK. Test to ensure that the system has reached a certain STAGE of CLEANLINESS
9 ordered by **PRIORITY**	1 2	Order the test scenarios/cases into various priority buckets so as to efficiently execute based on the situation.The focus here is efficiency of execution be it manual or automated. Test EFFICIENTLY in a given situation
10 mapped to **ENVIRONMENT**		In the case of system to support various environments(configurations) , map environment specific test scenarios and cases to the given environments. This comes in handy if we need to specifically evaluate impact of changes/fixes for specific environment. Test effectively for different BUSINESS ENVIRONMENTS

The Beauty of Organisation

"Ultimately, it's not just about doing the work. It's about satisfying our soul by finding beauty in what we do. Organising test assets isn't a dry or clinical exercise—it's about bringing life and aesthetics to even the drabbest scenarios."

Summary

Organising test scenarios and cases across the dimensions of product, test, and business aspects provides a comprehensive, multidimensional framework for evaluation. This method enhances clarity, promotes optimal effort, and transforms the testing process into a structured yet aesthetically satisfying endeavour. By integrating form and structure into the organisation of test assets, this approach not only ensures adequacy and efficiency but also elevates the overall experience of testing.

Perspective #10- Doing Less and Accomplishing More

This perspective, the first in the DOING series, delves into achieving more by doing less, focusing on four key strategies: avoiding unnecessary actions, addressing tasks early to reduce future effort, prioritising essentials, and leveraging tools for delegation.
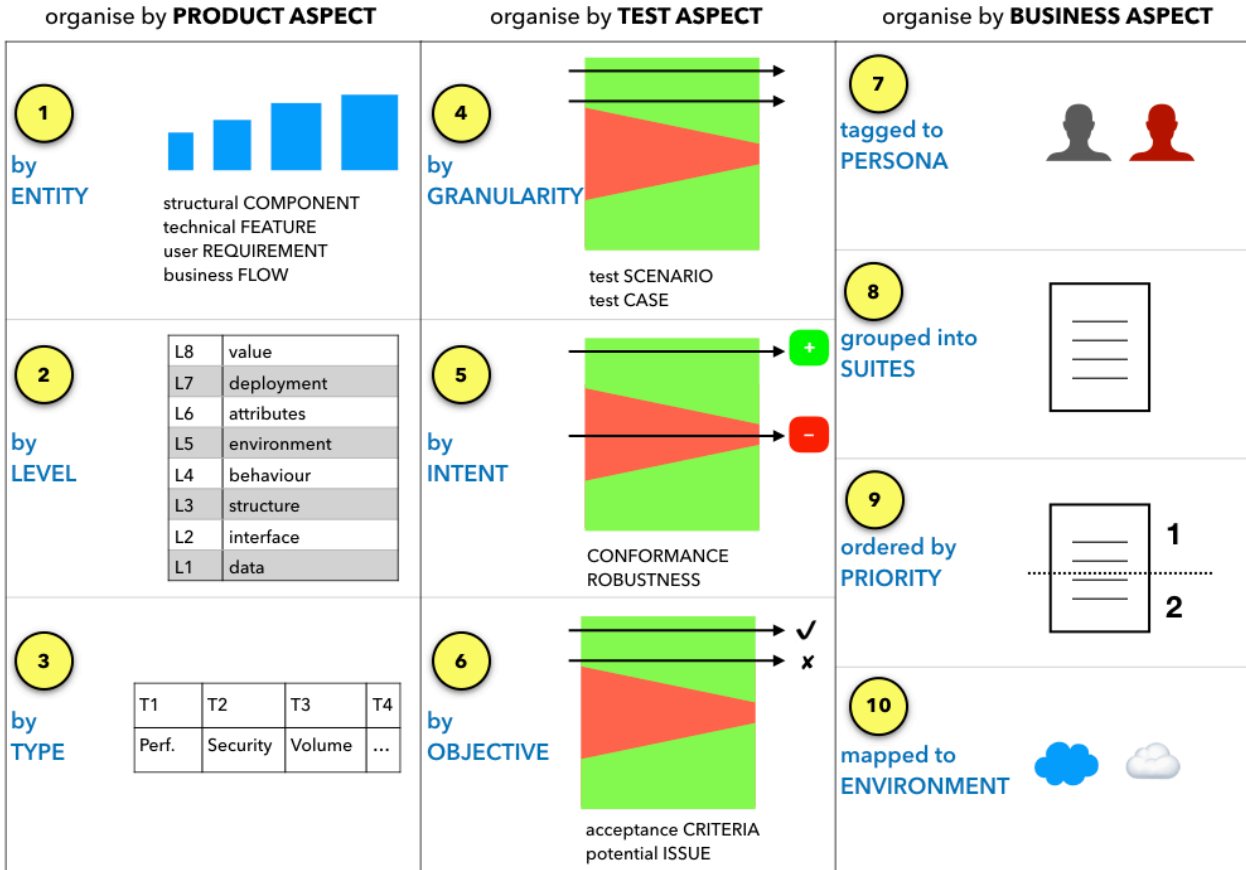
The Drive for Speed

In today's fast-paced world, there's an intense focus on speed. The need to run tests quickly, to fit frequent test cycles into compressed timelines, and to ensure that the quality of frequent builds and releases isn't compromised. This has placed a significant emphasis on automation. But is automation enough? As build and release cycles shorten, the hunger for speed grows, while end users' expectations for quality continue to rise.

So, how can we do less and accomplish more beyond automation? Here are four approaches:

Don't Do at All – Prevention

The first approach is about *prevention*: stopping issues from entering the system in the first place. This can be achieved by adopting a **fault-oriented mindset**, which involves being vigilant about potential defects that might creep in.

From the earliest stages of requirements formulation to design and coding, hypothesising about possible issues is crucial. Defensive programming plays a key role here, ensuring that the code is resilient and less prone to defects.

Do Less by Doing Early – Detection

The second approach involves detecting issues early to prevent them from snowballing into bigger problems. This can be achieved in two ways:

1. **Effective Development Testing**: Look for issues as they emerge during construction and integration phases.
2. **Focus on Testability**: Build hooks into the code to enable observation, analysis, and efficient evaluation using *smart checklists*.

By addressing potential problems as soon as they arise, we can mitigate the need for extensive corrections later.

Do Less by Doing Optimally – Optimised Detection

One challenge of ongoing updates—whether for new features, modified functionality, or bug fixes—is the potential to break previously working code. Regression testing, while necessary, can become a time-consuming and effort-intensive process.

Instead, let's consider *immunity*.

The Pesticide Paradox

Dr Boris Beizer, in his classic book *Software Testing Techniques*, illustrates the "pesticide paradox":*"A poor farmer loses his crop and is advised to use pesticide. After initial success, the pests become resistant, and new pesticides are required. The cycle repeats."*

Similarly, over time, software systems become "immune" to test cases—those tests no longer reveal defects. Each time a test passes, it provides valuable information about areas of the system that may have *hardened*. Use this insight to intelligently reduce redundant test cases.

Additionally, when issues of a similar kind appear throughout the system, statistical sampling techniques can be employed to select a representative subset of test cases, minimising effort without compromising coverage.

Do Less by Delegating to Tools – Automated Detection

Automation frees us from manual execution, allowing tests to run frequently and unattended as part of the build pipeline. However, to maintain automation's effectiveness, it's essential to synchronise test scripts with evolving code. Automation acts as a powerful health check, ensuring the system behaves as intended.

Yet, automation has its limitations. While it efficiently verifies existing functionality, it lacks the creativity to explore and uncover novel issues. This underscores the importance of balancing automated and manual testing efforts.

A Visual Framework: The When-How Axis

To conceptualise these approaches, imagine a two-dimensional axis:

- X-axis: **When** is it done?

- Y-axis: **How** much is being done?

The four quadrants formed represent the strategies of prevention, early detection, optimal detection, and automated detection.
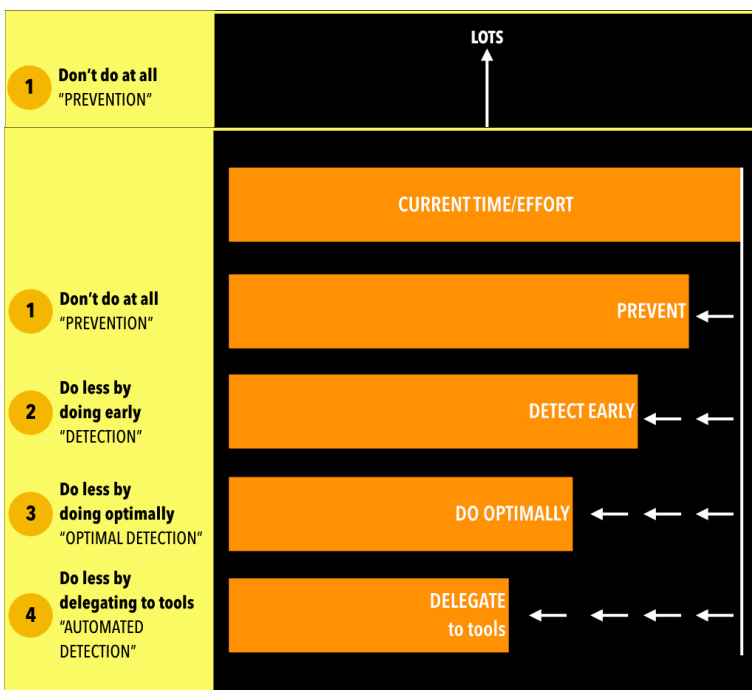
Summary

Test scenarios and cases are invaluable assets. When structured thoughtfully across product, test, and business dimensions, they provide multiple perspectives, enabling clarity and precision. A well-organised testing framework not only enhances efficiency but also elevates the aesthetic and intellectual satisfaction of the process, leading to brilliance in execution.

Perspective #11 - Move Rapidly by Doing Less

This perspective highlights how the act of execution can be accelerated using a smart checklist, which provides significant intellectual leverage in how we work.

The Essence of Efficiency: Lessons from Long-Distance Running

Long-distance running is not just about raw power, nutrition, or apparel; it is about great running form. It's about expending less energy to cover more distance, faster.

Running Form

"When we maintain good body position—head over shoulders, shoulders over hips, hips over the mid-foot upon landing, and arms swinging directly ahead—we run with good form and use less energy to run faster." – Brendan Cournane, *Good Running Form for Beginners*

Running well requires each part of the body to be strong enough to do its job and work in seamless coordination. This effortless harmony enables speed and endurance.

How does this translate to testing? Decompose the larger system-under-test into well-defined entities:

1. Structural Components – The building blocks

2. Technical Features – The elemental behavioural pieces

3. User Requirements – Meaningful functionality

4. Business Flows – The useful, real-world workflows

Focus on what issues to uncover in each of these areas. An earlier perspective r article, *The 4W to Structuring a Problem Well*, provides more detail. The diagram below depicts the system-under-test as a two-dimensional matrix: the x-axis represents type of entity, divided into strips of varying granularity, the y-axis represents different test types targeting specific issues.
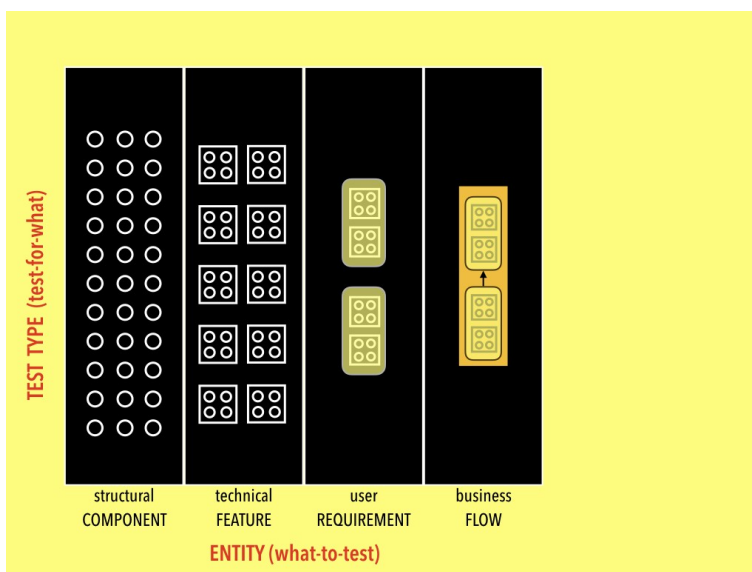
The Power of a Smart Checklist

To evaluate any entity, there are three key approaches:

1. **Human** – Develop scenarios and execute them manually.

2. Automated – Develop scenarios, convert into automated scripts, and execute them.

3. Human Aided by Smart Checklist – Use pre-designed checklist as a problem-solving aid, execute manually or automated when feasible.

A smart checklist is akin to good running form—it allows us to work effortlessly and effectively while maintaining speed. Why a checklist? Let's explore.
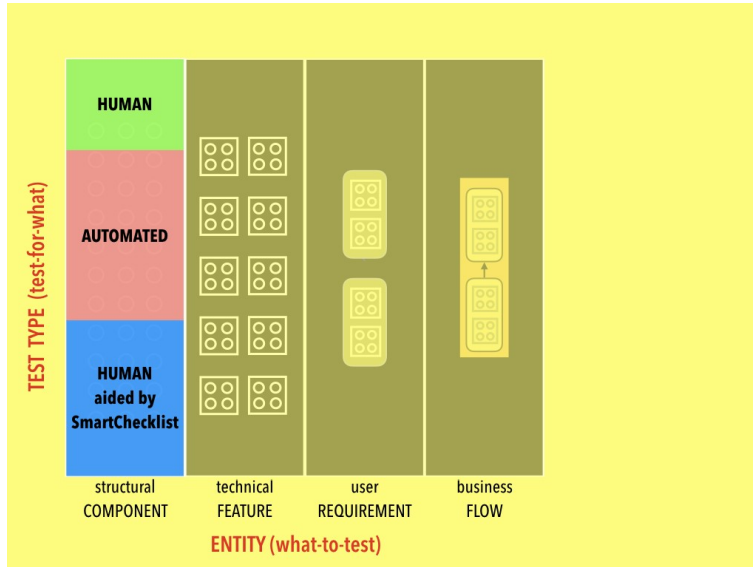
A Lesson from Aviation: Checklists Save Lives

On 30 October 1985, a massive plane carrying five times the typical payload of bombs crashed shortly after takeoff in Dayton, Ohio. The cited cause? Pilot error. A newspaper declared, "This was too much airplane for one man to fly." Boeing, the plane's manufacturer, almost went bankrupt.

The solution? A pilot's checklist. Flying the new plane was too complex to rely on memory alone, no matter how skilled the pilot. The result: 1.8 million miles flown without an accident.

In complex environments, experts face two main challenges:

1. Human Fallibility – Memory lapses, especially with routine tasks.

2. Step Skipping – Neglecting steps that seem insignificant in a complex process.

Checklists provide a safety net, instilling discipline and higher performance.

Smart Checklists: Thinking Tools, Not Box-Ticking

Atul Gawande, in *The Checklist Manifesto*, categorises problems into three types:

• Simple– Individual tasks with straightforward steps.

• Complicated– Tasks requiring coordination among multiple people or teams.

• Complex– Situations where the same inputs may yield different outcomes.

Smart checklists solve these problems by acting as cognitive nets, catching lapses in memory and attention. In mature disciplines like medicine, aviation, and construction, smart checklists have prevented countless errors, saved lives, and saved billions of dollars.

What Makes a Checklist "Smart"?

Smart checklists are not about mindless compliance or ticking boxes. They stimulate thinking and support fault-proofing. A smart checklist enables decision-making and responsible actions by providing:

1. **Compliance Checks** – Preventing critical oversights.

2. **Guidance** – Ensuring coordination and proper consideration.

3. **Stimuli for Analysis** – Encouraging deeper thinking and better decision-making.

For testers, this translates into designing checklists that combine simple, low-level compliance tasks with higher-order meta-scenarios that act as intellectual aids during test execution.
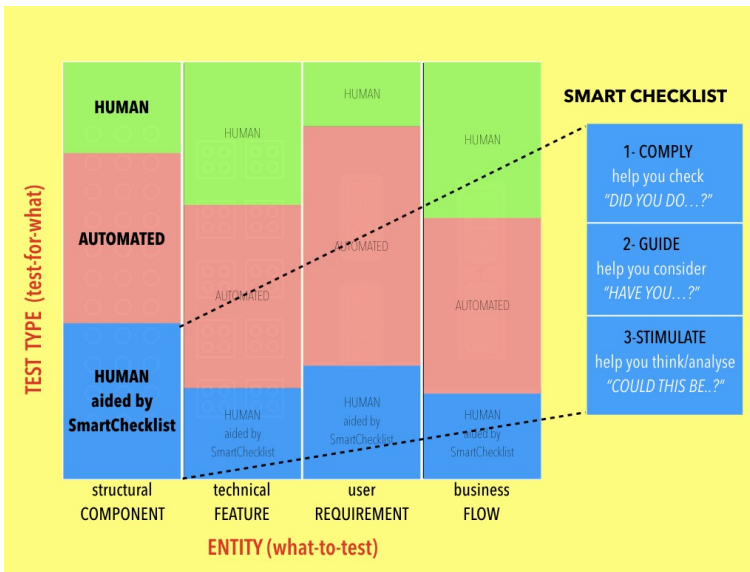
Enabling Speed Through Smart Design

Smart checklists accelerate execution by providing pre-formed designs, saving significant effort in crafting scenarios and enabling rapid focus. Depending on the type of entity under test, the balance between human, automated, and smart-checklist-aided testing may vary.
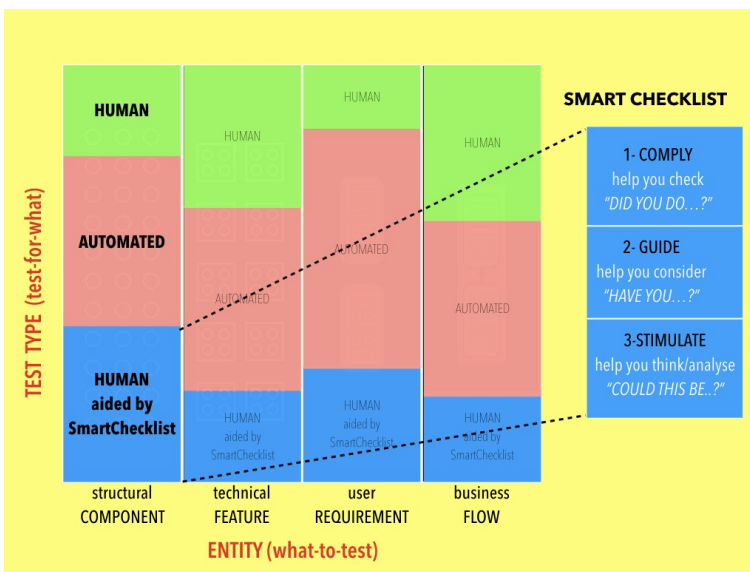
The Role of Tools: Humans and Machines

In today's world, where speed is paramount, tools are vital for testing activities, from environment setup to scenario execution and reporting. However, automation is not the only option for speed. Smart tools, like smart checklists, provide a simpler yet highly effective alternative to "move rapidly by doing less."

Summary

Testing involves three key phases: strategy formulation, scenario design, and execution. While we often focus on automating execution to save time, strategy and design can also benefit from smart checklists. Especially in scenarios where human involvement is necessary or automation is costly, smart checklists offer a practical, efficient solution to achieve speed without sacrificing quality.

Perspective #12 - Stay Engaged and Excel

Engagement, powered by mindfulness, can significantly enhance the way we approach rapid and immersive session-based testing.

The Evolution of Testing

In today's world of rapid, short increments, testing is best performed frequently and swiftly. Automation has become indispensable. Gone are the days of long testing cycles performed at the end of development. Modern testing thrives on nimble, effective iterations to keep up with continuous changes–be it fixes, modifications, or new additions.

Automated Tests: Vital Health Checks

As software continually evolves, it is essential to add new scenarios to the suite of automated tests that regularly exercise this extensive chain. These long chains essentially serve as health checks. While they may have previously identified bugs that were subsequently fixed, the likelihood of discovering entirely new types of bugs through these scenarios becomes increasingly remote. Rerunning them after every change helps ensure that existing functionality remains intact and system health is preserved.

However, the inherent malleability of software both its virtue and its challenge often introduces side effects that go unnoticed, leading to new breakages. This cycle of breaking, fixing, and reinforcing is a natural part of growth. It mirrors how we build our bodies: through intense exercise that creates micro-tears in muscles, followed by consuming protein to heal and strengthen them further. Interestingly, in software development, we are mimicking nature's principles of growth and evolution.

Rapidly Updating Test Suites

Once critical tests have been automated, how do we keep pace with the ever-changing software landscape? A solution lies in short, focused testing sessions. These sessions uncover new scenarios that can be executed manually or transformed into automated tests. Staying mindful and in the flow enhances this process tremendously.

The Power of Mindfulness and Flow

As a long-distance endurance cyclist, I discovered the magic of mindfulness by focusing on the front wheel rather than the mile marker during long rides that lasted for hours. It was about not worrying about the distance to the destination or judging my performance by how far I had travelled, but simply enjoying the act of cycling. The magic lies in being present i.e. mindful, not in the past or the future.

Want to experiment with mindfulness? Take a glass of water, sip it slowly, and focus solely on the act of drinking. Feel the water caress your lips, swirl around your tongue, and quietly descend down your throat. Spend a few moments fully immersed in this simple experience, and you will find enjoyment as time seems to stand still. Wouldn't you want a testing session to feel just like this?

Lessons from a 1,000 km Cycling Journey

Let me tell you a story about my 1000 km nonstop cycling challenge, where the maximum allowed time to complete was 75 hours, with the clock never stopping. I finished it in under three days. Riding a bicycle continuously for three days in heat, wind, cold, climbs, and rain is incredibly demanding. It requires both strength and fitness. Long hours against unrelenting wind and challenging climbs demand a strong mind filled with positive energy. Most importantly, it calls for an extreme focus on the NOW, to be mindful without worrying about the cutoff times in the future. Stay in the present, pedal strong, and enjoy the journey. Of course, a high-tech bike and excellent cycling techniques that conserve energy help immensely.

This is illustrated in the picture below.

As a cyclist, I view the achievement of covering long distances in less time with minimal effort not as an act of brute force but as one requiring the mind\u2019s power to be effectively harnessed. It about staying in the moment, living each second fully, unmindful of the future, and in doing so, generating more power to go the distance.

I discovered that when I became mindful, my power output was immense. My legs moved in rapid, circular motions without pain, with a quiet breathing rhythm. The wind stopped being a bother, the hills and climbs became inviting, and the speedometer stayed consistently high. To me, this was a "Wow" moment. This is when I truly discovered the power of mindfulness.

Mindfulness in Testing

Now let us relate this to the discussion at hand. When expectations increase, the scope of testing expands, and it seems we need to acquire more skills (much like building physical strength). Testers must be adept in the domain to step into the end user's shoes and skilled in technology and tools to fully exploit automation. This relates to "Skills - What we know," which is undeniably important.\n\nThe natural progression is to leverage technology to test faster. This means injecting probes into the system for better testing, automating as much as possible, delegating repetitive work to machines, and increasing process agility to focus on critical tasks. This is "How we leverage" technology, tools, and processes.

By becoming more skilled and exploiting technology effectively, we can accelerate and achieve more. But is that enough? Staying focused on the present during a test session works wonders. It allows us to be mindful, unleashing creativity and enabling skills to
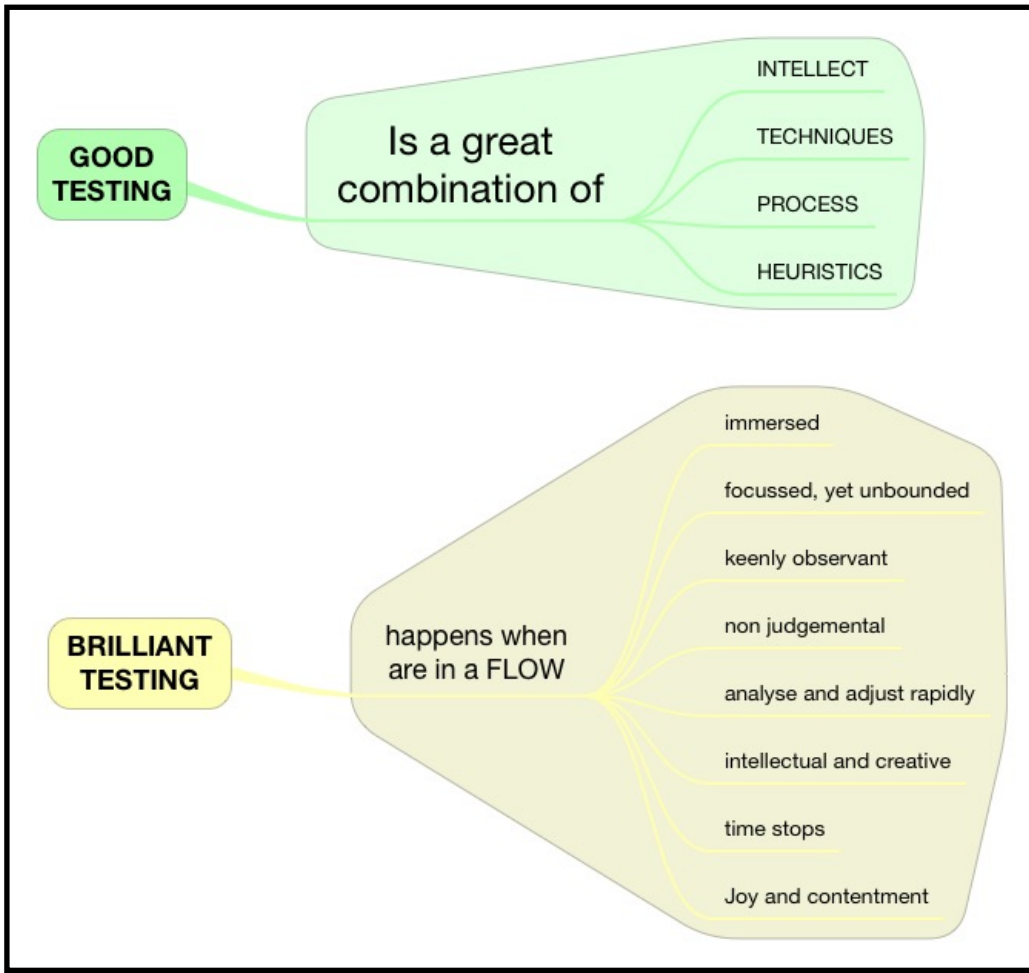
flow naturally. This requires letting go concerns about deadlines and the future. It is about immersing yourself in the current test session of staying in the context, absorbing it, and using it well.

It is about freezing the present.

A short, mindful session of testing can be thought of as this equation: $P = \text{Lt}_{t \to 0} W/t$, where P is productivity, W is the work to be done, and t is the time taken. When we effectively freeze time through mindfulness, our productivity skyrockets. This means being conscious of the present moment, staying focused, and enjoying the session. Mindfulness transforms testing from a task into something enjoyable and fulfilling.

As engineers, we rely on good techniques, tools, processes, and intellect to perform well. But how can we do even better, perhaps brilliantly? Once we have exhausted all "external" resources, it's time to tap into the immense "internal" potential we all possess.

This is about going beyond the intellectual mind and delving deeper into the subconscious to harness its infinite potential. A great starting point is achieving a state of mindfulness the flow.

Beyond Tools: Harnessing Internal Potential

After leveraging tools, techniques, processes, and intellect, it's time to explore the vast internal potential we all possess. Enter the state of flow—a place where energy harmonises, work feels effortless, and performance peaks.

What is Flow?

Flow is the state of immersion where:

- Energy is fluidly harnessed.
- Work feels effortless yet brilliant.
- Observations are sharp and dispassionate.
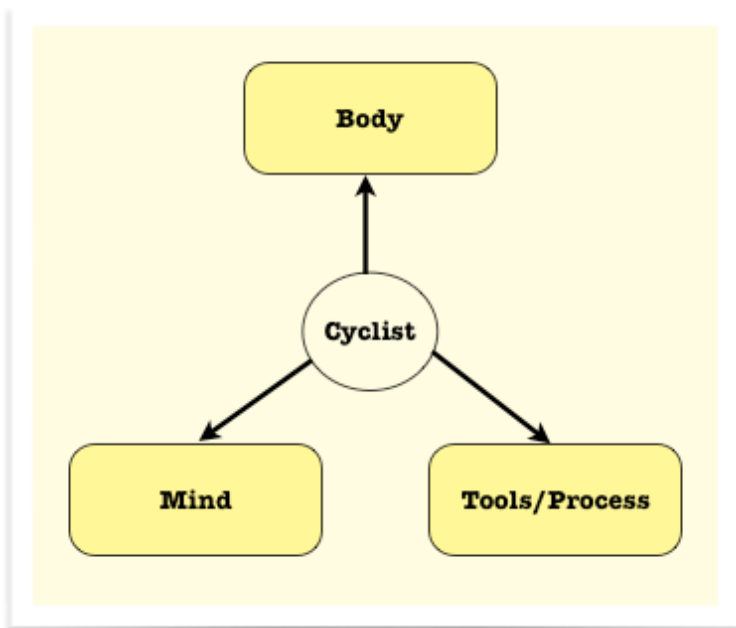
- Actions are agile and precise.
- Time seems frozen, enabling immense productivity.

Achieving Flow in Testing

Flow emerges when multiple sensory inputs converge harmoniously:

- Visual: Use visual text, sketch maps, or mind maps to understand systems, strategise, and record observations.

- Tactile: Engage the sense of touch by writing or drawing freely with pen and paper.

- Auditory: Quietly talk to yourself or play calming background music.

Short, focused sessions (45-90 minutes) with clear objectives are key. Engage deeply and let the flow take over.
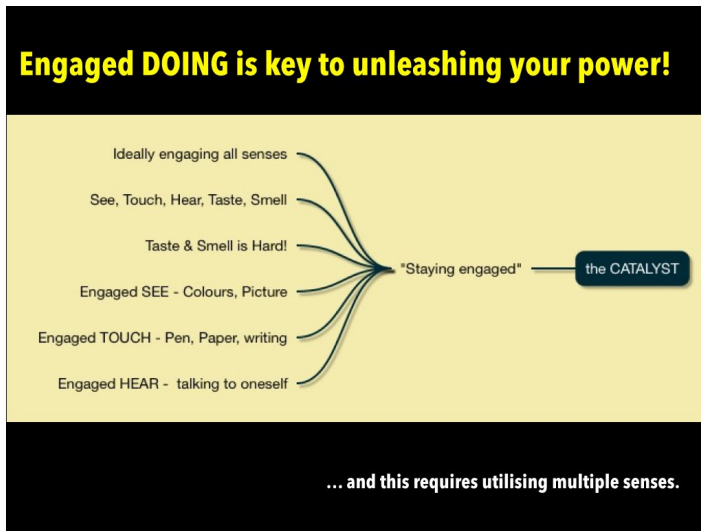
Benefits of Flow

When in flow, logical thinking merges with creativity, enabling expansive, non-judgemental thought. Observations sharpen, ideas emerge rapidly, and testing becomes both effective and enjoyable. This approach Hypothesis-Based Immersive Session Testing (HyBIST) consists of three phases:

1. **Reconnaissance:** Rapid understanding and baseline setting.

2. **Exploration:** Context-driven strategising and designing.

3. **Recoup:** Retrospective analysis for insights and progress.

Summary

Automation and machine intelligence enhance productivity and capabilities, but unlocking our inner potential is essential to complement these advancements. By embracing mindfulness and achieving a state of flow, we can maximise creativity and performance. Hypothesis-Based Immersive Session Testing (HyBIST) methodology refines testing into a focused, mindful practice that synergises with automation, empowering engineers to work more efficiently and effectively.

"We are SmartQA evangelists. For over two decades we have transformed how individuals, teams and organisations have practised testing. We espouse methodology to test intelligently. Our mission - Elevate to high performance via SmartQA."
www.stagsoftware.com

The HyBIST Approach to SmartQA - MASTERCLASS
Testing is deep probing to seek clarity and in the process uncover, preempt issues rapidly. The HyBIST approach enables designing smart probes and probing the system smartly.
https://smartqa.academy/courses/smartqa-using-hybist

doSmartQA - AI based Smart Probing Assistant to interrogate, hypothesise issues, design & evaluate user story or a set of stories in a sprint rapidly. An assistant for smart session-based testing based on HyBIST.
Download personal edition from <u>here</u>

SmartQA Musings - A gentle flurry of interesting thoughts on smart assurance as a weekly webcast. A refreshing view of assurance to broaden & deepen thoughts/actions.
www.stagsoftware.com/subscribe

SmartQA Biweekly - Ignite your curiosity with fresh insights, thought-provoking ideas, and inspiring content delivered straight to your inbox every fortnight.
www.stagsoftware.com/subscribe

# A rich collection of original content on smart assurance

SmartQA eBooks - for Sr Engg/QA managers, Sr Test Practitioner & Young Test Practitioners too.

50 Tips for SmartQA

Communicate Clearly

do SmartQA -The HyBIST Approach

HyBIST at a glance

Download from www.stagsoftware.com/smartqa-ebooks

SmartQA Wisdom - Profound nuggets of wisdom to think deeply, do rapidly & smartly for Test Practitioners.

SmartQA Wisdom

on
Smart
Assurance

Thiruvengadam Ashok
STAG Software
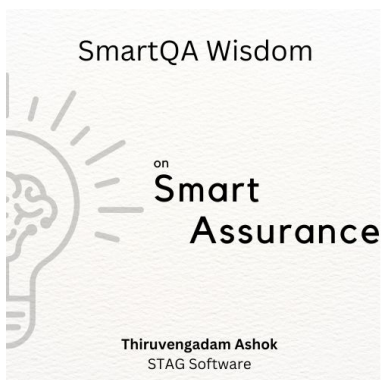
on Smart Assurance
on Personal Growth
on Test Design
on Smart Understanding
on Mindset & Habits
on Problem Solving

Download from www.stagsoftware.com/smartqa-wisdom