TÉCNICO
UNIVERSIDADE
DE LISBOA

Governed Human-AI Collaboration Across the Software Development Lifecycle

A Process Framework and Its Reference Implementation

Tomás Taborda - ist1103641

Supervisor: Prof. Miguel Mira da Silva

Outline

1 Introduction

2 Research Methodology

3 Systematic Literature Review

4 Research Problem

5 Research Proposal

6 Implementation

7 Demonstration

8 Evaluation

9 Conclusion

1. Introduction

Adoption has outpaced method: the bottleneck shifts to validation

84%
developers use or plan to use AI

61%
functionally correct (one leading agent)

+20%
believed faster (experienced developers)

+98%
PRs merged (1,255 teams)

29%
trust its accuracy

10.5%
secure

-19%
measured slower in an RCT

+91%
review time, no delivery gain

Stack Overflow, Zhao et al., METR, Faros AI (2025)

2. Research Methodology

Systematic Literature Review

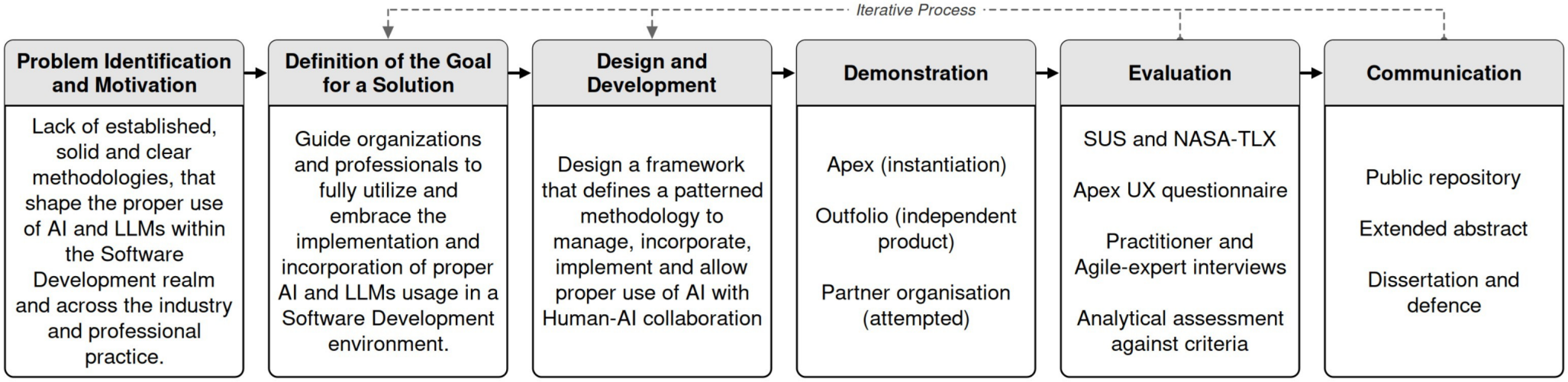
Kitchenham: plan, conduct, report

Grounds the research problem

Design Science Research

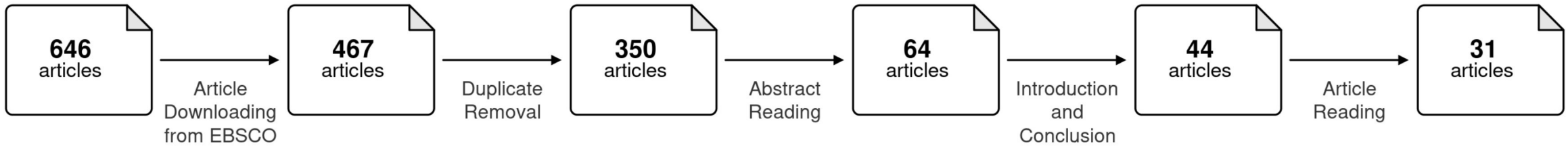
Peffers et al.: six activities

Builds and evaluates the framework and Apex



Iterative: building Apex fed back into the framework

3. Systematic Literature Review



RQ1	RQ2	RQ3	RQ4	RQ5
Benefits	Consequences	Methodologies	Stakeholder opinions	Industry sectors

Every benefit carries a matching risk

Quality and productivity are top benefits; cost, reliability and security top consequences

Human oversight is the strongest shared position

Stakeholder groups diverge, but converge on oversight (7 studies)

No accepted framework for the whole lifecycle

Prompt-level techniques and HITL exist, but no integration framework

4. Research Problem

No structured guidance for AI across the SDLC with oversight, quality, security and accountability

Objective: a process framework with AI as a first-class collaborator: bounded, validated and traced

The problem can be separated into four dimensions:

1	2	3	4
Placement	**Governance**	**Realisation**	**Practitioner support**
AI per phase	accountable control	implementable in tools	usable by teams

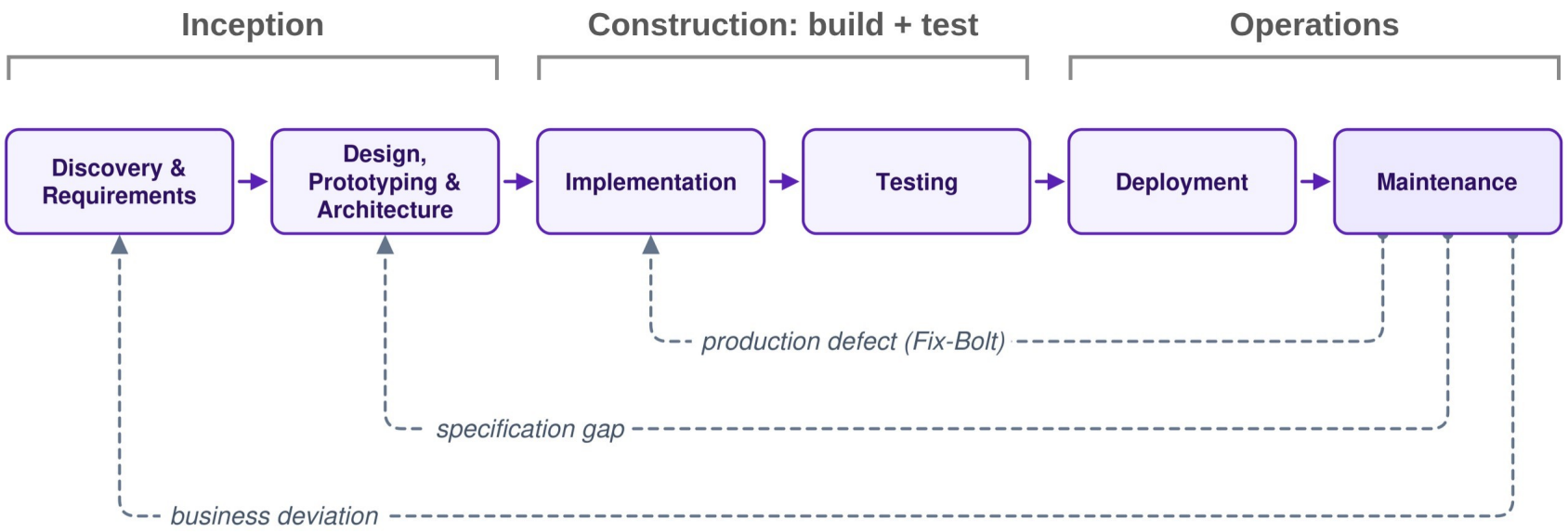
5. Research Proposal - Framework

AI drafts → **A hat decides** → **A gate closes the phase**

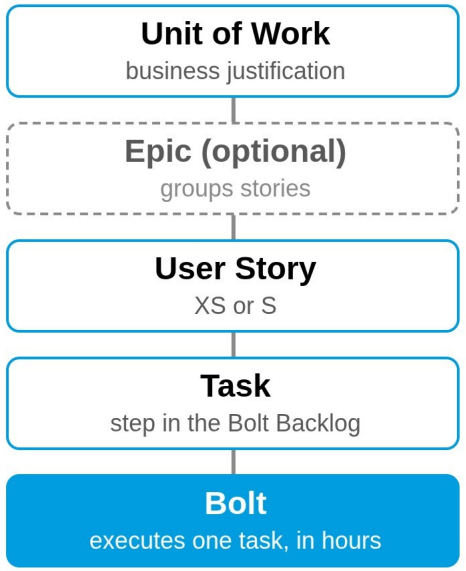
Strategic Alignment precedes six phases; five gates; every loop-back is governed by the specification

Every phase works from a versioned spec, global plus per story, locked at gates and amended only on record

AWS AI-DLC phases



Work hierarchy



Adopted from AI-DLC:
Bolt, Unit of Work, Mob Elaboration

Extended: Testing decoupled into its own phase, avoiding circular validation; Maintenance classifies every signal behind a diagnostic firewall

5. Research Proposal - Phase by Phase

Phase	AI drafts	Hat that owns the gate	If the gate fails
Strategic Alignment	candidate objectives and KPIs	PM	Unit of Work does not reach the Squad
Discovery	small User Stories, then Gherkin	Trio: PO, Tech Lead, Design Lead	back to Mob Elaboration; oversized stories split
Design	UI structure and technical specification	Design Lead + Tech Lead	no task enters the Bolt Backlog
Implementation	code within the Consistency Factor	Developer: must explain the logic	tasks back to the Bolt Backlog
Testing	BDD scripts from the approved Gherkin	QA, in a separate environment	Fix-Bolt with the exact failing evidence
Deployment	IaC, configuration, migrations if needed	Whoever deploys	objection fed back; release does not proceed
Maintenance	narrow diagnosis and a bounded fix	Developer or PM classifies	no gate: loop-back to the owning phase

Bolt Master owns no gate: logistics stays apart from technical authority. **Solo work** self-checks, but the evidence must still exist.

5. Governance Mechanisms

Hats

Functions, not job titles; every decision owned

Gates

Owner, evidence, consequence; failure routes work back

Spec-Anchored Continuity

Versioned reference, changed only by amendments

Bolts

One task, hours long; continuous flow, no sprints

3 metrics · safety and flow, not output volume

Bolt Cycle Time

time per task, from pull to done; tests continuous flow

Context Traceability Rate

share of deployed stories traced back to their specification

AI Defect Escape Rate

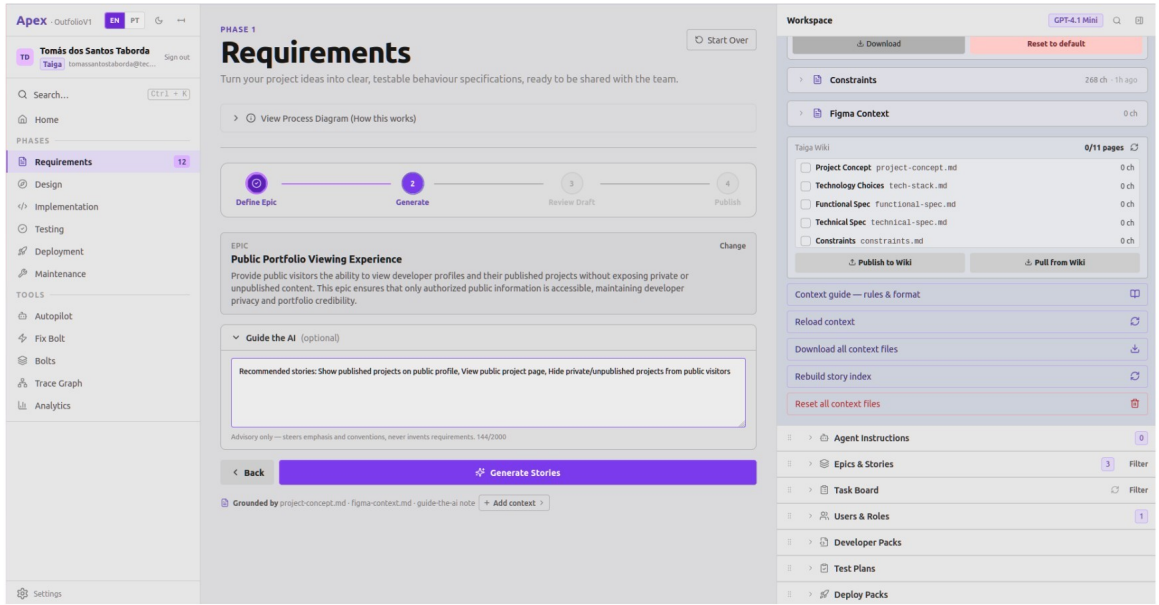
deployed stories with a confirmed post-deployment bug; tests the gates

7 design principles behind every mechanism

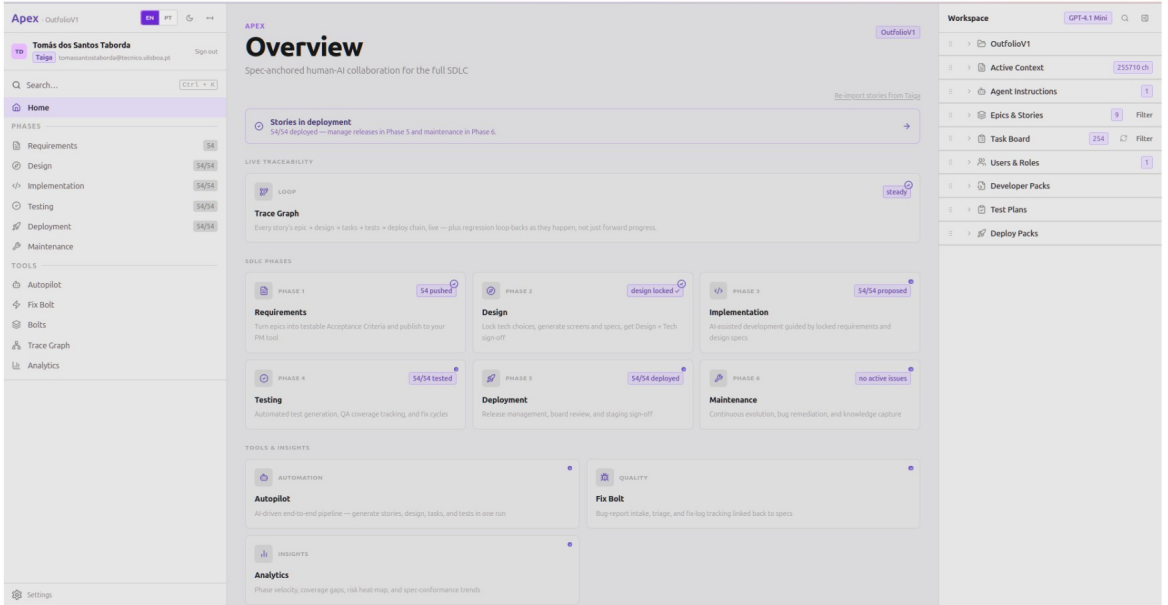
Human-in-the-Loop by Design · AI as Augmentation, Not Substitution · Process-Level Governance · SDLC-Wide, Phase-Aware Integration · Functional Accountability · Risk-Proportional Validation · Spec-Anchored Continuity

6. Implementation - Apex

Not a coding agent: Apex governs whether code still answers its requirement



Phase 1: Requirements



Overview: six phases, live traceability

Stable spec ID per artefact	**Spec files, not a database**	**Taiga and Plane adapters**	**Anthropic, OpenAI, Google**

...and much more

6. Implementation - Realisation

10 / 12

constructs realised as specified

1. Six lifecycle phases
2. Quality gates
3. Spec-Anchored Continuity
4. Stable spec IDs
5. Interactive Bridge
6. Consistency Factor
7. Bolts and Fix-Bolt
8. Bolt Cycle Time
9. Context Traceability Rate
10. **AI Defect Escape Rate**
11. **Spec-drift detection**
12. Server-side reconciliation

1 proxy · 1 partial

Realised does not mean reliable

AI Defect Escape Rate

proxy

proxy by design: sees only reported, linked bugs

Spec-drift detection

partial

one project-wide edit flagged 44 stories; reworked per story

Bolt Cycle Time

fixed

measured gate transitions, not task time, until fixed

Practitioner feedback: DevOps role removed; dual sign-off replaced by recorded evidence

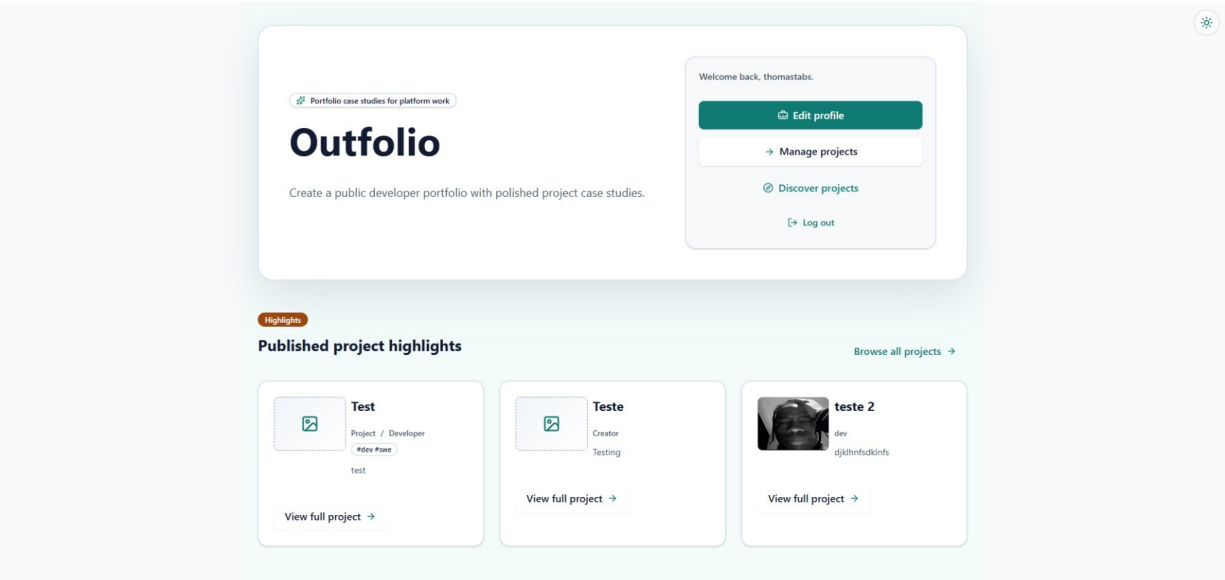
Quality assurance: ~1,638 backend, 318 frontend unit and 24 end-to-end tests

7. Demonstration - Outfolio

1 Apex
feasibility: the instantiation itself

2 Outfolio
full lifecycle, independent product

3 Partner organisation
attempted, never engaged



Outfolio turns OutSystems apps into public case studies

Goal: show the constructs can be operationalised

Pre-registered; unfavourable results get equal weight; the author wore every hat

9 epics	54 stories deployed
9 days, 14 to 22 Sep	123 commits
422 log entries	62 test files

7. Demonstration - Under Real Failure

Production database wiped

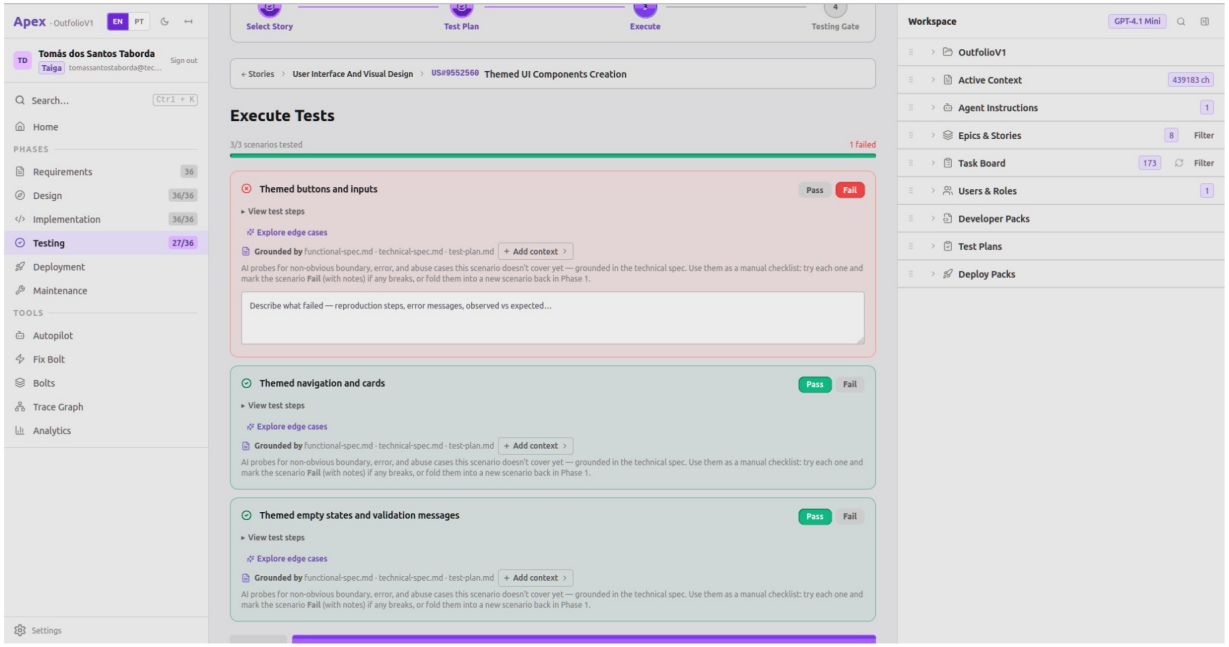
agent reset script, inside a phase: no gate could catch it

Defect hidden by mocks

all 36 stories passed only after real failures; manual probing found it

+18 stories via loop-backs

12 then 6, each a recorded decision, classified before any code



Apex Phase 4 Testing: one failed scenario, two passed

7. Demonstration - Observations

0.22 h
median Bolt Cycle Time,
254 tasks

92.6%
Context Traceability
50 of 54 stories

98.65%
Spec Conformance

1.85%
AI defect escape
weak proxy

⚠ 6 points where the framework fell short

1. **Destructive action inside a phase**
2. **Testing Gate passing obsolete tests**
3. **Defects only live testing found**
4. **Confidently wrong recommendations**
5. **Blurred phase boundaries**
6. **Responsibilities skipped**

Conclusion: the constructs can be operationalised under real failure, but governance at phase boundaries cannot see everything.

8. Evaluation - Framework

Analytical assessment

Five criteria fixed before results

Met

- SDLC alignment
- Human-in-the-loop

Partially met

- Clarity
- Completeness
- Governance

Shortfalls: unstated per-story assumption, defect proxy, stale context reported with high confidence

Interviews

5 practitioners and 1 Agile expert; T1-T5 and E1 analysed

Key Points:

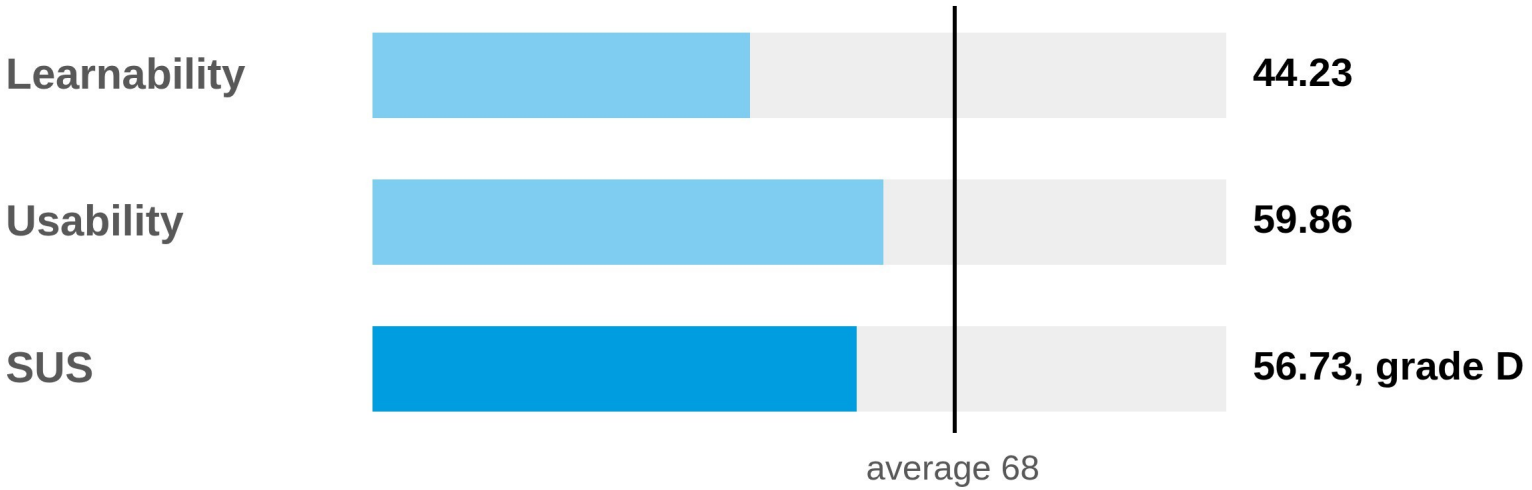
- **Spec as durable project memory**
  T2
- **Executable tool makes the method legible**
  E1
- **Risk of linear, waterfall-like rigidity**
  T4, anticipated by E1
- **Spec as source of truth divides opinion regarding scalability**
  T2 vs Section 7.5

⚠ Gate comprehension not measured
⚠ Demonstration run by its designer

Conclusion: no criterion is unmet, but linearity is the main concern and the gates are the least evidenced construct.

8. Evaluation - Apex Instantiation

13 participants · 9 tasks · mostly junior, unmoderated



Difficulty is first contact, not continued use

9 of 13 cited density or navigation, not the AI mechanisms

Trust: looks calibrated (median 4 of 5), yet one participant trusted the output without reviewing it in detail

NASA-TLX

21.5 / 100

workload, comparatively low

heaviest: Design task, mental 47.7

Limits

- ⚠ Small, mostly junior sample
- ⚠ Nothing recorded
- ⚠ Delivered quality not measured

Note: SUS and NASA-TLX measure the tool, not the method; a weak score is not evidence against the framework.

9. Conclusions

1
SLR synthesis
31 studies, 5 RQs

2
Framework
6 phases, 5 gates, 3 metrics

3
Apex
10 of 12 constructs realised

4
Evaluation
method vs tool claims apart

Governed Human-AI collaboration can be specified, implemented, exercised under real failure and evaluated.

Answering the research problem across its four dimensions: placement, governance, realisation and practitioner support

Future work

Independent teams
multiple organisations, separate hat holders

Other PM tools
test tool independence in practice

Direct metrics
production telemetry in place of proxies

AI cost governance
token and compute cost per Bolt

Design-phase support
the heaviest task for participants

First-contact usability
treated as a non-negotiable requirement

TÉCNICO
UNIVERSIDADE
DE LISBOA

Thank you!

Questions and discussion

Tomás Taborda - ist1103641

Supervisor: Prof. Miguel Mira da Silva

Apex
apex-bolt.com

Outfolio
outfolio-v1.vercel.app

Documentation
github.com/thomastabs/apex