Verification Capability Benchmarking Service

Mike Bartley, Alpinum

Do you recognise any of these issues?

ALPINUM CONSULTING

- Why do we always miss our verification deadlines?
- Should we have found these bugs earlier?
- Why are there still bugs in basic use cases?
- Why do our verification teams achieve different quality levels?
- How best to ramp up these new verification engineers
- Why do we seem to repeat the same mistakes?

Why benchmark?

- To understand the current verification capability
  - and identify improvements
- Better prepare for tomorrow
  - Increasing verification complexity
  - Reduced time to market
  - Reducing costs
- How does benchmarking help with that?
  - Measure the maturity of functional verification activities
  - Gain an integrated view of the organisation’s functional verification capability
  - A framework for continuous process improvement
    - Define goals, priorities and actions
    - Regular measurement of progress

Other benchmarks are available

- CMMi
  - General-purpose and heavyweight, and not verification specific
- Evolving Capabilities Model (Foster and Warner)
- DV-CMM
  - How is DV-CMM different?
    - View of the whole org from a verification perspective
    - Objective measure
    - Framework for process improvement
    - Top-down decomposition and bottom-up evaluation
    - 3 key elements: capability, maturity and process
  - DV-CMM is a proven lightweight benchmarking process
    - Proven on 5+ organisations
    - With 10+ sites
    - And 25+ projects
    - Now extended to include AI capabilities

Process areas

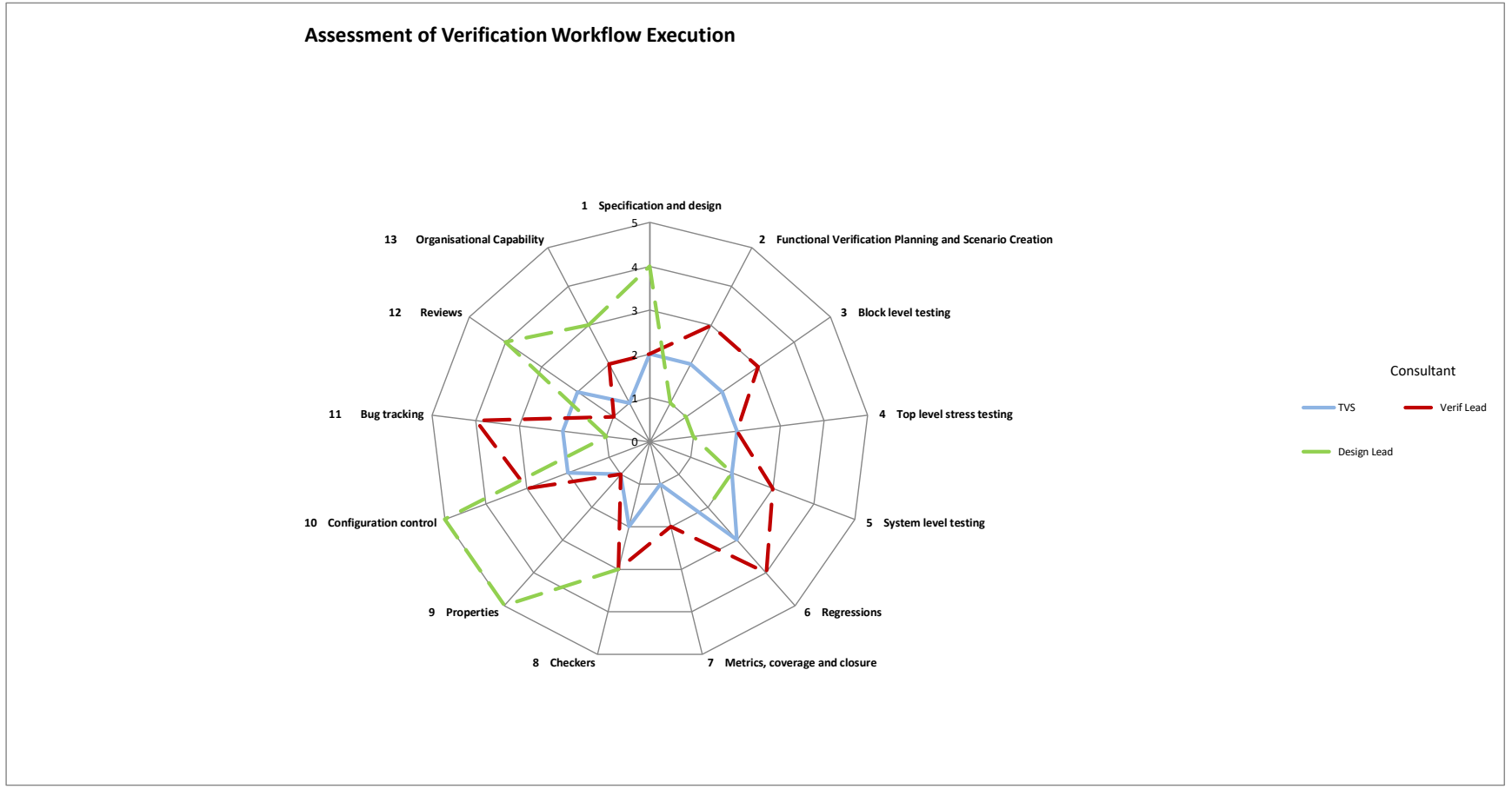
1. Specification and design
2. Verification Planning and Scenario
3. Block level verification
4. Subsystem verification
5. System-level verification
6. Regressions
7. Metrics, coverage and closure
8. Checkers and properties
9. Configuration control
10. Debug
11. Bug Tracking
12. Reviews
13. Organisational Capability
14. NEW: AI adoption

Evaluation: Axes and levels

ALPINUM CONSULTING

	Initial	Managed	Defined	Quantitative	Optimising
Ownership	Individual	Project Team	Project Stakeholders or ad hoc groups of projects	Community	Company wide or institutionalised
Visibility	Not documented No reviews. No metrics.	Documents incomplete or unmaintained. Point reviews. Progress metrics.	Maintained docs. Continuous tracking against quality metrics.	Living docs. Quantified quality metrics.	Data integrated across the organisation.
Execution	Ad hoc	Tasks performed but completion not explicitly checked	Tasks planned and implemented in a systematic fashion. Check completion of planned tasks.	Quantifiable metrics used for coverage closure and release determinism	Quantifiable metrics used to drive continuous improvement.

Different Views of Verification Capabilities



Recording the analysis

FV-CMM process areas	Maturity	Ownership	Visibility	Execution
5 System level testing	3. Defined	2. Project Team	3. Maintained documents and point reviews	3. Tasks planned and implemented in a systematic fashion
5.1 The purpose of each test bench should be clearly identified	3. Defined	2. Project Team	3. Maintained documents and point reviews	3. Tasks planned and implemented in a systematic fashion
5.1.1. The purpose and the scenarios to be reached by each test bench should be clearly identified. The purpose must consider the appropriate level of testing for the various scenarios (e.g. integration with other IP, software debug features, low power features, performance validation via benchmarking)	Environment to run real-world software. This is the big thing emulators give them, and it hits stuff they wouldn't find anywhere else. A mix of what is historically available (Symbian, WinCE and Linux), what feels as though it could be helpful to and the available simulation capacity. Use irritators for OS booting and stress apps. Try to make use of some key system features such as virtualisation and TrustZone. Some reusable software like "crashme", "memcopy". Run this against different configs of hardware such as a small L2 cache to increase stress. Can also use Cambridge knowledge from A9 of what cases found bugs.			
5.1.2. Regression testing, using appropriate scenarios and checkers, should be used to validate bug fixes and ensure errors are never reintroduced.				

1

2

3

4

1. Topic
2. Maturity levels for this topic
3. Sub-topic with maturity levels
4. Details

How does benchmarking provide answers?

Question	How benchmarking helps
Why do we always miss our verification deadlines?	Weakness in particular process areas
Should we have found these bugs earlier?	Is system verification stronger than block and/or top?
Why are there still bugs in basic use cases?	Weak verification planning and reviews
Why do our verification teams achieve different quality levels?	Organisational capabilities fail to promote knowledge sharing.
How best to ramp up these new verification engineers	First understand their strengths and areas for improvement
Why do we seem to repeat the same mistakes?	Are you collecting the right data? Are you doing continuous improvement via benchmarking?

Summary

- Benchmarking helps to
  - Measure the maturity of functional verification activities
  - Gain an integrated view of the organisation’s design verification capability
  - A framework for continuous process improvement
- DV-CMM is a proven lightweight benchmarking process
  - Now extended to include AI capabilities

Thank You!

CONTACT US TODAY:

Mike Bartley, Founder/CEO, Alpinum Consulting

mike@alpinumconsulting.com

+44 (0) 7796 307958

Visit our website

Book a call

LinkedIn