How I Learned to Stop Worrying and Love Benchmarking Functional Verification!

DVCon 2012
Mike Bartley, TVS
Recognise any of these?

• Why do we always miss our verification deadlines?
• Surely we could have found these bugs earlier?
• How comes we seem to have bugs in some basic use case scenarios?
• Why do our sites have such different verification capabilities?
• How do I integrate this new team in ....?
• Why do we seem to make the same mistakes over and over again?
Why benchmark?

• To understand 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 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
  - Does not address the specific capabilities relevant to verification

- **Evolving Capabilities Model**
  - Foster and Warner

- **How is FV-CMM different?**
  - View of the whole org from functional verif aspect
  - Objective measure
  - Framework for process improvement
  - Top-down decomposition and bottom-up evaluation
  - 3 key elements: capability, maturity and process
Different Views of Verification Within a Project

Self Assessment of Verification Workflow Execution

1. Specification and design
2. Functional Verification Planning and Scenario Creation
3. Block level testing
4. Top level stress testing
5. System level testing
6. Regressions
7. Metrics, coverage and closure
8. Checkers
9. Properties
10. Configuration control
11. Bug tracking
12. Reviews
13. Organisational Capability

TVS
Verif Lead
Design Lead

Test and Verification Solutions 28th Feb 2012
Process areas

1. Specification and design
2. Functional Verification Planning and Scenario
3. Block level
4. Top level stress testing
5. System level
6. Regressions
7. Metrics, coverage and closure
8. Checkers and properties
9. Configuration control
10. Debug
11. Bug Tracking
12. Reviews
13. Organisational Capability
Verification Teams Can Have Wildly Different Views

Verif Team Self Assessment of Verification Workflow Visibility

- Specification and design
- Functional Verification Planning and Scenario Creation
- Block level testing
- Top level stress testing
- System level testing
- Regressions
- Metrics, coverage and closure
- Checkers
- Properties
- Configuration control
- Reviews
- Organisational Capability

- Verif Mgr
- Verif Eng 1
- Verif Eng 2
- Verif Eng 3
## Evaluation: Axes and levels

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

Looking at Different Sites Across the Organisation

TVS Assessment of Different Sites

1. Specification and design
2. Functional Verification Planning and Scenario Creation
3. Block level testing
4. Top level stress testing
5. System level testing
6. Regressions
7. Metrics, coverage and closure
8. Checkers
9. Properties
10. Configuration control
11. Bug tracking
12. Reviews
13. Organisational Capability

TVS Site 1
TVS Site 2
TVS Site 3
The Benchmarking Process

Environment to run real world software. This is the big thing emulators gives them and it hits things they wouldn't find anywhere else. A mix of what historically available (Symbian, WinCE and Linux), what feels as though it could be useful and the available simulation capacity. Use irritators for OS booting and stress apps. that try to make use of some key system features such as virtualisation and TrustZone. Some reusable software like "crashme", "memcpy". 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.

Regression testing, using appropriate scenarios and checkers, should be used to validate bug fixes and ensure errors are never reintroduced.
So how does benchmarking answer these?

- Why do we always miss our verification deadlines?
  - Weakness in particular process areas

- Surely we could have found these bugs earlier?
  - Is system verification stronger than block and/or top?

- How comes we seem to have bugs in some basic use case scenarios?
  - Weak verification planning and reviews
So how does benchmarking answer these?

• Why do our sites have such different verification capabilities?
  – Weak organisational capabilities do not promote knowledge sharing

• How do I integrate this new team in ....?  
  – First understand their strengths and areas for improvement

• Why do we seem to make the same mistakes over and over again?  
  – 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 functional verification capability
  – A framework for continuous process improvement

• FV-CMM is proven lightweight benchmarking process