What is Inspection?

- A structured process for finding and fixing defects
- Used to remove defects as early in development as possible
- A *simplified* model:

```
Roles
Activities
Products
```

```
1. Planning
2. Detection
3. Collection
4. Correction
```

- Planning Form
- Defect Report Form
- Defect Collection Form
- Defect Correction Form
- Corrected Artifact

- organizer
- inspector
- moderator
- inspectors
- author
- author

© 2011 Fraunhofer Center, Maryland
Why Inspection?

- A long history of research & application shows that structured human inspection is one of the most cost-effective practices for achieving quality software:
  - “Cost savings rule”: Cost to find & fix software defects is about 100x more expensive after delivery than in early lifecycle phases, for certain types of defects.
    - IBM: 117:1 between code and use
    - Toshiba: 137:1 between pre- and post-shipment
    - Data Analysis Center for Software: 100:1

- “Inspection effectiveness rule”: Reviews and inspections find over 50% of the defects in an artifact, regardless of the lifecycle phase applied.
  - 50-70% across many companies (Laitenberger)
  - 64% on large projects at Harris GCSD (Elliott)
  - 60% in PSP design/code reviews (Roy)
  - 50-95%, rising with increased discipline (O’Neill)
  - … many others
Problem Statement

• System development is often decomposed to handle complexity.

• Software increasingly plays a larger role in the system...
  • Research on system hazards in NASA's Constellation Program revealed that 51% of the hazards contained at least one software cause [Basili et al., 2010]

• … but it is still just one part of the system
  • Assurance activities are often conducted independently.
  • Domain knowledge may affect quality of activities.
  • Need a more integrated approach → inspection across the system.
    • For each inspection, consider a holistic view of the system.
Our proposed approach

• Research goal: Provide guidance for teams on planning and conducting inspections across a system.
  • Non-intrusive
  • Cost-effective
  • Adaptable

• Philosophy: Package best practices, including adapting principles from software engineering.

• Our context is inspections of highly critical systems
  • But should be generalizable to other domains.

Health Check – Inspection Process Assessment Methodology
The “Process Health Check”

• Assess the current inspection process – standards and policies against practice.
• Provide best practices and guidelines for defining an inspection process.
• Identify areas that could benefit from recommendation.
The “Process Health Check”

- Assists with integrating an inspection into the larger system or CE lifecycle
- Used during project planning
- Has implications for how inspection preparation is carried out
Methodology – Overview

• Create baseline of best practices.
• Package best practices in a framework.
• Continuously refine framework:
  • Proof of concept study.
  • Pilot Study
  • Deployment of the approach.
Building Baseline – Sources

- Understand the practices for system inspections:
  - Sources:
    - NASA, DOD, ESA standards and handbooks
    - System engineering literature.
  - Well known software best practices
    - NASA, ESA, DOD, RUP, literature
  - Source re-elaboration:
    - Understanding the real issues and needs
      - System is different from software
    - Definition of a common taxonomy
      - Different standards can use different taxonomies
    - Gathering and merging best practices
      - Different standards and practices can propose different solutions
Building A Baseline – Triggering Questions

- What techniques do people use to review system/software quality issues during development?
  - Which artifacts serve as input to these techniques?
  - Which techniques account for both systems and software?
- How do system engineers and software engineers participate in each other’s activities?
  - Should they participate in each other’s activities? How? When?
- Is there any similarity between software inspections and system reviews?
  - How can our knowledge and experiences in software inspection help to improve the system review process?
Exploring Interactions between Software and System

- Reviews are “Key Decision Points” in both system and software development.
- Reference models allow us to define system and software reviews that:
  - Reason about *types of information* and how it is encapsulated in documentation at various phases ➔ What’s available as input?
  - Understand issues of timing, coordination, and communication across subsystems ➔ How do we assure that future activities can be done correctly?
Formulating Recommendations

• For each review type, reference models allow us to reason about:
  • Structure of the review
    • Team composition and expertise.
    • Amount of material to inspect.
    • Meeting length.
  • Artifacts to be inspected
    • Type and notation of documents.
  • Quality attributes
    • Mandatory and optional attributes.
    • Which expertise should be checking which qualities.
    • Which artifacts are appropriate for checking various qualities.
Formulating Recommendations

• For each review type, reference models allow us to reason about:
  • Structure of the review
    • Team composition and expertise.
    • Amount of material to inspect.
    • Meeting length.

These parameters have been shown to affect effectiveness of (software) inspection.

There are heuristics available.

Did they stand the test of time?
Formulating Recommendations – Inspection Structure

• Our recommendations are tested against a database of inspection results from across NASA centers.
  • 2500+ inspections
  • 5 Centers
• We unified, scrubbed, and verified the data
  • Sparseness: Not all inspections collected our metrics of interest
    • E.g. 721 reported # inspectors
    • E.g. 627 reported page rate
  • Outliers: We retained extreme values that used same definition of the metrics, if not of an inspection
    • E.g. Page rates of hundreds of pages per hour
    • E.g. Meeting length of less than 30 minutes
• Defect data is sensitive – Raw data can be used by us but cannot be shared with other teams
Formulating Recommendations – Inspection Structure

- Work at NASA in the mid-90s by Dr. John Kelly identified heuristics for key parameters (moderator’s control metrics), e.g.:

<table>
<thead>
<tr>
<th>Team size:</th>
<th>Page rate:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Too small – miss important expertise</td>
<td></td>
</tr>
<tr>
<td>Too large – drive up costs, dampen discussion</td>
<td></td>
</tr>
<tr>
<td>=&gt; Rule of thumb = 4 to 6</td>
<td></td>
</tr>
<tr>
<td>Too small – miss interrelations</td>
<td></td>
</tr>
<tr>
<td>Too large – thorough review impossible</td>
<td></td>
</tr>
<tr>
<td>=&gt; Rule of thumb = 10 to 30 pgs for reqts, 20 to 40 pages for test plans, etc.</td>
<td></td>
</tr>
</tbody>
</table>

- Our database confirms that heuristics are still good predictors of inspections with most defects found.

<table>
<thead>
<tr>
<th>Team size:</th>
<th>Page rate:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg results for all projects:</td>
<td></td>
</tr>
<tr>
<td>If followed: 14 defects detected</td>
<td></td>
</tr>
<tr>
<td>If not: 7 defects detected</td>
<td></td>
</tr>
<tr>
<td>Significant, p&lt;0.0005</td>
<td></td>
</tr>
<tr>
<td>Avg results for all projects:</td>
<td></td>
</tr>
<tr>
<td>If followed: 14 defects detected</td>
<td></td>
</tr>
<tr>
<td>If not: 6.5 defects detected</td>
<td></td>
</tr>
<tr>
<td>Significant, p&lt;0.0005</td>
<td></td>
</tr>
</tbody>
</table>

- Yet, fewer projects are able to follow them:

| Team size: 10% of contemporary projects followed |
| Page rate: 15% of contemporary projects followed |
Formulating Recommendations – Inspection Structure

Page rate = 20
Original heuristic (avg = 15.4)

Page rate = 40
Maximize number of defects (avg = 13.1)

Maximize defects found per hour

Design
Packaging Best Practices – as Assessment Process

• Assessment questions and (best practice/recommendation) answers about:
  • Development and review process.
    • Development model, amount of material to inspect, meeting length.
  • Review team
    • Team composition and expertise.
  • Artifacts to be inspected and produced
    • Type and notation of documents.
    • Inspection metrics
  • Quality attributes
    • Mandatory and optional attributes.
    • Which expertise should be checking which qualities.
    • Which artifacts are appropriate for checking various qualities.
  • Context questions: understand the need for tailoring of the best practices.

• Assessment questions to tie the recommendations to project context – development process, etc.
Health Check Process – An Informal Model

1. Provides
2. Examines
3. Asks follow up questions
4. Gives
5. Examines
5. Consults
2. Consults

Red flags (i.e. deviation from expectation) may lead to:
- Recommendations to the inspection process
- Updates to the health check Q-A's

Process Documents

Structures, artifacts, Quality attributes

20 sets of Q's & A's
Health Check Process – Example of Assessment Question

• High-level question:
  • Who are the team members that are generally required to participate in a review of a particular artifact?

• Best practice recommendation:
  • In most types of reviews, an inspection team should represent at least the following perspectives: requirements/user, integration and implementation, quality and process assurance

• Detailed-level/probing questions (if mismatch occurs):
  • If a recommended team member is missing from the actual review team, what is the reason for this omission? Who performs his/her tasks in the actual review team?
  • If a member of the actual review team is missing from our recommended team composition, why is this particular member needed? Who performs his/her tasks in the recommended review team?
Proof of Concept – Application of Health Check

• Applied with NASA team developing safety-critical hardware interlocks.

• Assessment Process:
  • Step 1: Team sends us process documentation.
    • Development and assurance process.
  • Step 2: Gather answers to the health check questions, and compare them against the expected answers.
  • Step 3:
    • Ask follow-up questions
    • Formulate recommendations.
  • Step 4: Analyze feedback.
Proof of Concept – Application of Health Check

• Recommendations:
  • Issue 1: No inspection is req. in requirements phase
    • Recommendation: A review should be performed during requirements phase, perhaps based on our SRR checklists
  • Issue 2: V&V Matrix is only constructed during design phase.
    • Recommendation: V&V matrix is based on requirements. It is a valuable artifact for SRR. Move its development earlier in the lifecycle.
  • Issue 3: Development and evolution of test plan is not clear.
    • Recommendation: Test plan is valuable artifact for every type of review. Test plan could be created in the early lifecycle phases.
  • Issue 4: SRD and SSRD are input to the design and implementation phase, but no change or request document are shown as outputs
    • Recommendation: It is beneficial to be open to look for requirement problems even in the later phases of development. Note explicitly constraints that disallow changes to such documents.
Future and Ongoing Work (1)

- Further validate and refine our approaches:
  - Reaching out to teams who would be interested in applying health check and providing feedback.
    - Currently work with a NASA team looking at certification review from both software and hardware side.
- Further extend our approaches for inspecting complex electronic applications.
  - Understand the interface between CE and System.
  - Understand which phase of CE is more closely related to software and which phase is more related to hardware.
Ongoing Work (2)

• Expand best practices recommendations to other V&V technologies
  • Assess trade-offs of each V&V technique and formulate an assurance strategy based on combination and/or sequences of techniques.
Acknowledgement

• This work was sponsored by a grant from NASA’s Software Assurance Research Program (SARP), “Inspections for Systems and Software.”

• Contact us:
  • Madeline Diep
    mdiep@fc-md.umd.edu
  
  • Forrest Shull
    fshull@fc-md.umd.edu