How to do Inspections When There is No Time

Terry Shepard **Diane Kelly**
Dept. of Electrical and Computer Engineering
Royal Military College of Canada
Kingston, Ontario
Canada, K7K 7B4
(613)541-6000
x6031 x6491
shepard@rmc.ca kelly-d@rmc.ca

ABSTRACT
The concepts of inspection are presented, organized around the three Ms of inspection (Management, Mechanics and Metrics). Fundamentally, inspection provides a way of observing and quantifying software processes while improving software quality. It can be used as a statistical quality control technique for software production. The tutorial addresses variations in inspection processes and techniques, and presents recent research results that investigate which variations are more effective than others. This is done with recognition of the need to adapt inspection approaches to fit local circumstances.

Keywords
software inspection, Fagan-style inspection, scenario-based reading, task-directed inspections, active design review

1 INTRODUCTION

Research says software inspections are an effective, essential part of software development. Yet, according to some industry practitioners, inspections are difficult, costly, ineffective, and excessively time consuming. So what's gone wrong?

Developed over 25 years ago, Fagan style code inspections are still the most widely practiced form of inspection. Given the pace of today's software development, there is a perceived need for other forms that are lighter weight, more flexible, and make better use of precious resources. Useful new forms have been introduced and validated in industry. We will look at a variety of these.

2 GOALS OF THE TUTORIAL

This tutorial aims to provide inspection practitioners and developers with a range of approaches for designing software inspections to suit their own software development environment. One of the most common perceptions is that there is no time to do inspections, even though there is a payback on resources invested in inspections. In this tutorial, we look at designing your inspection process and finding novel ways to shift resources to inspection. We consider issues from three perspectives: management, mechanics, and metrics. The tutorial provides a toolkit of ideas to help you answer questions such as: What aspects of inspection should you pay special attention to? How can you match resources to your inspection goals? What makes the biggest difference to inspection effectiveness? The end result will help you design or streamline your inspection process to fit your needs, whether personal, small group or large group.

3 PRESENTATIONS

The tutorial starts with an overview of current industry inspection practice, giving the novice background on what inspections are about. We introduce the three Ms of inspection (Management, Mechanics, Metrics) as a basis for tailoring inspections to your own environment, including variations on inspection techniques and processes which will be of interest to experienced inspection practitioners as well as to novices.

Management addresses those concerns that influence the success of inspections, including culture, goal-setting, resource issues, and process improvement choices.

Mechanics cover both the process and techniques needed to design software inspections for a given environment. Processes vary from Fagan's original version of inspection, with participant roles, inspection meetings, and follow-up activities, to the light-weight inspection activities of extreme programming. Inspection techniques vary from unstructured ad-hoc approaches to the systematic, specific, and distinct characteristics of structured approaches such as scenario-based reading. We discuss important research and improvements suggested by a number of authors, including Votta [15] and Chernak [3]. Votta [15] suggests that not every inspection needs a meeting. Chernak [3] discusses

0-7695-1050-7/01 $10.00

concerns when using checklists as a basis for inspection. Material for the tutorial is drawn from all the references below ([1] to [17]).

Metrics are core to inspection. If no metrics are taken, the activity is better referred to as review or walkthrough. The metrics that you choose to include in your inspection process have to be meaningful, usable, and useful. To ensure that this continues to be true, metrics may change as a project progresses. Metrics are a necessary part of feedback to management, and a necessary part of process improvement.

We present a case study of an inspection exercise designed to address a particular situation in industry, including the management goals, the process and techniques developed to address the goals and constraints, and the results. This particular case study has led to continuing research at the Royal Military College into the task-directed inspection technique.

The conclusion summarizes the main points and provides possible approaches to evolving inspection in a software development environment with significant challenges.

Opportunities for questions and discussion will be provided throughout, so the tutorial will in part be a workshop.

REFERENCES

1. Kent Beck, Extreme Programming Explained: Culture Change, Addison Wesley, 1999
2. Jarir K Chaar, Michael J. Halliday, Inderpal S. Bhandari, Ram Chillarege, "In-Process Evaluation for Software Inspection and Test", IEEE Transactions on Software Engineering, Vol 19, no 11, Nov. 1993, pp 1055-1070
3. Y. Chernak; "A Statistical Approach to the Inspection Checklist Formal Synthesis and Improvement"; IEEE Trans. on Software Engineering, Vol 22, No. 12, Dec. 1996, pp.866-874
4. Tom Gilb and Dorothy Graham, Software Inspection, Addison Wesley, 1993. see also http://www.result-planning.com/
5. Robert B. Grady, Successful Software Process Improvement, Prentice Hall, 1997
6. Phillip M. Johnson; "Reengineering Inspection", Communications of the ACM, Feb. 1998, Vol 41, No 2, pp 49-52
7. Diane Kelly and Terry Shepard; "Task-Directed Software Inspection Technique: An Experiment and Case Study", IBM CASCON, Toronto, Nov. 2000
8. J.C. Knight, E.A. Myers; "An Improved Inspection Technique", Communications of the ACM, Nov. 1993, Vol 36, No 11, pp. 51-61
9. Laitenberger, Oliver Laitenberger, Khaled El Emam, Thomas Harbich; "An Internally Replicated Quasi-Experimental Comparison of Checklist and Perspective-Base Reading of Code Documents"; [online], http://www.iese.fhg.de/network/ISERN/pub/technical_reports/isern-99-01.pdf; 1999
10. David L. Parnas, David M. Weiss; "Active Design Reviews: Principles and Practice", Proceedings 8th International Conference on Software Engineering, Aug. 1985
11. Adam A. Porter, Lawrence G. Votta, Victor R. Basili; "Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment", IEEE Trans. on Software Engineering, Vol 21, No 6, June 1995, pp 563-575
12. Adam A. Porter, Harvey P. Siy, Carol A. Toman, Lawrence G. Votta; "An experiment to Assess the Cost-Benefits of Code Inspctions in Large Scale Software Development", IEEE Trans. on Software Engineering, Vol 23, No 6, June 1997, pp 329-346
13. Glen W. Russell, "Experience with Inspection in Ultralarge-Scale Developments", IEEE Software, January 1991, pp. 25-31
14. University of Maryland, Notes on Perspective Based Scenarios; [online]; http://www.cs.umd.edu/projects/SoftEng/ESEG/manual/pbr_package/node8.html; Nov. 1999
15. Lawrence G. Votta; "Does Every Inspection Need a Meeting?", SIGSOFT'93 - Proceedings of 1st ACM SIGSOFT Symposium on Software Development Engineering, ACM Press, New York, 1993, pp 107-114
16. Lawrence G. Votta; "Does the Modern Code Inspection Have Value?"; Presentation at the NRC Seminar on Measuring Success: Empirical Studies of Software Engineering; March 1999; [online], http://www.cser.ca/seminar/ESSE/slides/ESSE_Votta.pdf
17. David A. Wheeler, Bill Brykczynski, and Reginald N. Meeson Jr., Software Inspection: An Industry Best Practice, IEEE CS Press, 1996