

Applying Small-Investment, High-Return Review Techniques for Fast-Paced Teams

Laura Faulkner
Applied Research Laboratories
The University of Texas at Austin

Abstract

Multiple methods exist for performing formal inspections, reviews and walkthroughs. Many of these face obstacles in the fast-paced world of software development because they are rigid, time-consuming, and require numerous players and roles. The methods and numbers of individuals involved in such sessions create complexity by geometrically increasing the interpersonal interactions as inspection group size increases, and by requiring extensive pre-inspection preparation time and post-inspection interpretive and reporting activities. Using organizational dynamics, cognitive psychology, and drawing on real-world inspections experience, the author distinguishes multiple streamlining strategies while maintaining optimal quality, proposing a Structured Review approach that takes the best of all worlds and makes for a fast, low-impact, high-return review process.

Introduction

Values of inspections, reviews, walk-throughs in software development are known by quality practitioners and others throughout the field. Peer reviews catch defects early, make for easier and cheaper testing, fixes and deployment (Pressman, 1997). There are pronounced differences between formal inspections, informal reviews, and this author has termed, “Structured Reviews.” A formal inspection is a highly defined process requiring multiple individuals, advanced preparation time for each person attending, adherence to specified roles within the meeting, and detailed record-keeping procedures. While that process is effective in maintaining order in the meeting, finding defects, and establishing boundaries for reaction and response, it is also cumbersome for small or fast-paced development environments. Accordingly, it is frequently abandoned or simply not applied in the first place. In contrast, informal reviews can provide a simpler method for such teams, but their lack of definition can render them both inefficient and ineffective for defect identification. An alternative approach is “The Structured Review”. The structured review is a combination of the most simple and effective elements of each of the other techniques. Its greatest impact can be found in applying it under the right conditions and in a simple way.

The Structured Review

The Structured Review consists of the following primary elements: 1) a pre-defined checklist that uses simple standards to identify defects that may arise in a particular phase; 2) a small group that can work quickly due to a reduced number of interpersonal interactions; and 3) peer responses unlimited by specific roles.

The elements of the method are effective in several ways. A checklist guide, written in the form of questions rather than lists of facts, creates *memory triggers* that facilitate real-time evaluation and reduce or eliminate the need for pre-meeting, individual reviews. The guidelines, rapid during-session approach, and limited hours needed to perform a Structured Review, all help

individuals maintain *cognitive attention*. In many human endeavors, after performing a certain task for period of time, a “leveling off” occurs in the ability to focus and continue noticing effectively. Further, the small, non-role-assigned group creates an atmosphere of intimacy, efficiency, trust which can increase communication and make it easier for the author or developer to accept feedback.

Benefits of a Structured Review Method

The Structured Review contributes to savings in time, frustration and cost, while contributing to the ability of individuals to identify defects in a peer review setting.

Preparation. Inspections are preparation-heavy. In contrast, the Structured Method applies a generic set of heuristics that does not have to be re-authored in preparation for each review. Accordingly, the Structured Review walks the reviewers through the process in real-time, eliminating pre-reviews.

Defect Identification. Reviews can miss numerous defects because they are not sufficiently detailed, and do not contain cognitive triggers for what is important. The Structured Method resolves this by providing specific items on which to focus and which serve as reminders to reviewers.

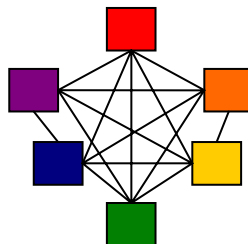
Time Investment. To perform a formal inspection, or even an informal review, teams must schedule review time prior to the inspection to effectively find defects. The reason for this is that there are insufficient guidelines for use during the inspection or review session. The Structured Method resolves this by providing pre-defined items that increase review efficiency, allowing for real-time evaluation and defect identification, without the need for extensive pre-meeting reviews.

Applying the Method

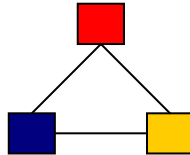
The overall process consists of the following steps:

1. Adopt or create a recording form: checklist criteria questions, with yes or no boxes (a comment column may also be added, but should be used in a limited fashion, as described below). Note that this need only be created once, then applied to multiple projects; and the criteria can best be derived from existing sources, eliminating the need for “from scratch” authorship.
2. Perform the reviews with *three people*, one of whom is the author/programmer
3. Only *identify* defects; do not use the review time to solve them.
4. Verify that identified issues have been addressed or repaired

Reducing Complexity: Three People, No Roles. The number of interpersonal relationships that exist in a 6-person meeting is 15.



The number of interpersonal relationships that exist in a three-person meeting is only three.



A smaller group makes for more efficient operation for several reasons. First, the reduction of personal interactions, as described and illustrated above, allows faster communications and increased input from the individuals involved. Additionally, fewer people makes it is less intimidating to both participants and especially the designer. This creates an atmosphere of greater intimacy and trust than can be achieved in a larger meeting. The elimination of roles in the Structured Review, means the meeting can begin immediately without multiple individuals having to settle into specific roles. Further, the persons within the group can interact without the constraints of roles required in a formal inspection, increasing simplicity of the process, allowing greater focus on the task of defect-finding rather than the tasks of specific roles.

Reducing Complexity: Checklist Defect Recording

The “checklist” for a Structured Review consists of a pre-defined criteria list in the form of questions that can be answered with a “yes” or “no.” This facilitates fast identification of defects, and helps groups avoid having their reviews deteriorate into time-consuming problem-solving sessions.

Parameters and Guidelines for the Meeting. The following recommendations can help the review session run smoothly and contribute to the focus and effectiveness of the group:

- a. Review small chunks of code, segments of requirements, etc.
- b. Schedule 1-2 hours (shorter is better)
- c. Use the first 5 minutes to determine whether to review line-by-line or as a whole
- d. Read aloud
- e. Work in chunks - draw break-point lines beneath each section of the product
- f. Take a 5-minute break at each break-point line
- g. Reiterate the criteria after each break
- h. Answer only the questions in the checklist

- i. Keep each other honest—agree between the 3 of you to stay with the procedures

On-Track Reviews

The effectiveness of the method stems from both its structure, and how it is applied.

Staying Focused. Without structure, review and inspection sessions can easily deteriorate to style issues. In a session where style issues dominate, substantive defects are not being found. This is a ready occurrence because style issues are easier to recognize than defects. Also, style issues can hook into individual opinions. This can generate personal investment, making them attractive for lively or heated discussion. The Structured Review resolves this by providing specific questions that do not address style, and a group process that avoids deviating from the questions.

Maintaining Efficiency. Inspections and reviews spill over into a “solution fest.” Even when style issues, and their inevitable solutions, are avoided, it can be tempting for a group to begin discussing solutions immediately when defects are found. The brain simply wants to go down the solution path; completing the path facilitates personal comfort by thoroughly finishing a thought so that it can be deleted and the brain can move on. However, if the individuals fully understand the purpose and task in a Structured Review session, efficiency can be increased, and personal discomfort avoided. The knowledge that “completing the task” means merely marking the defects, and the physical act of marking them can give individuals a sense of completion. Further, because simply marking the defects is easier than creating resolution, it can be even more attractive, in the context of a review meeting, than problem-solving.

Ease of a Single Approach. When to do an ‘inspection,’ a ‘review,’ or a ‘walkthrough’; abandoning all for less than optimal methods. Multiple methods are difficult to apply in the fast-paced software development field. A Structured Review method can be applied throughout the lifecycle of a project; if adopted as the usual method it can provide a single, easy way to perform effective reviews.

A Matter of Course: Structured Reviews. Adopt or write a set of heuristics for each type of review, for example, requirements, code, documentation (see “References and Resources” below). In project scheduling and planning, include a structured review with each small chunk of work to be performed. Further, let your team know that this is something to be done, simply as a matter of course.

Applying Reviews in the Lifecycle: Significant Areas for Return-on-Investment. Structured Reviews can be applied to multiple products in a development lifecycle. Also, each takes only about two hours to complete, and requires only three people to perform. This six-hour, full-time-equivalent commitment is low impact on both schedules and budgets. In contrast, a formal inspections may require, for example, eight individuals, four hours of preparation, and four hours of meeting time, for a full-time-equivalent commitment of sixty-four hours! While the potential for detailed, formal reviews to catch defects is high, and the cost-effectiveness of early fixes is known, the smaller, up-front investment of a Structured Review can increase ROI by lowering the “investment” figure in the equation.

Following is a list of phases/products to which a Structured Review can easily be applied. Phases marked with an asterisk (*) denote those in which a review can produce the most significant return. Four include a brief listing of sample criteria questions which could appear in Structured Review checklist:

1. Schedule

2. Requirements*. Some examples of requirements criteria questions include:

- a. Does the requirement explain “what” rather than “how”?
- b. Can it be broken up into more than one requirement?
- c. Is it testable? How will you know it has been fulfilled?
- d. Does it meet user needs?
- e. Does it conflict with any other requirements?
- f. Are things named consistently?

3. Design. Sample design criteria questions include:

- a. Does this design meet one or more requirements?
- b. Will it perform the required functions?
- c. Is the subcomponent design consistent with other subcomponent designs?
- d. Is it complete? Is there anything missing?

4. Code*. A sample of Q&A criteria for a code review may include:

- a. Are all logic algorithms, data structures, and calls within each module valid?
- b. Are data names descriptive?
- c. Does each line conform to the coding language?
- d. Is it complete? Is there anything missing?

5. Usability. One advantage of a usability checklist review, is that you can obtain usability information whether or not you have a usability expert in the review. The following is a small sampling of usability criteria that can be used in a Structured Review:

- a. Does information mean the same thing throughout?
- b. Are terms and language the ones that users will know and understand?
- c. Do things appear in the order in which the user would perform them?
- d. Do error messages propose a solution to the problem?

6. Documentation*

Achieving Closure. As with other types of inspection and review, to complete the process the author/developer commits to a repair date for all defects found; and the other two reviewers check back on that date to verify resolution. The report to project can be a simple accounting of defects found, and defects removed. If needed, a “solutions meeting” can be scheduled with the specific purpose of brainstorming solutions for the defects found; understanding that this is a possible later action makes it easier to simply identify defects during the review, and makes both meetings more efficient because each is focused on specific type of action and result.

Conclusion

Reviews and inspections to find defects up front are cost effective. Each of the three methods discussed have advantages and their own built-in costs. Inspections find the most defects. Reviews are easier than inspections. Selection of the appropriate method will depend on the timelines, budgets and goals of the project, and the potential effects of errors in the end product.

Structured reviews can be a healthy combination of ease and effectiveness, using rapid, inexpensive methods with the benefits of identifying defects for repair at a high rate of return.

References and Resources

The following sources were either used in the preparation of this paper, or will serve as resources to the practitioner:

- Faulkner, L.L., (2001). "Measuring Software Usability: Developing a Heuristic Evaluation Process and Tool," in *Proceedings*, International Conference on Practical Software Quality Techniques, April 2001, Orlando, Florida.
- Gilb, T., Graham, D. (1993). *Software inspection*. London: Addison-Wesley.
- McConnell, S. (1993). *Code complete: a practical handbook of software construction*. Redmond, Washington: Microsoft Press.
- McConnell, S. (1996). *Rapid development: taming wild software schedules*. Redmond, Washington: Microsoft Press.
- Pressman, R. (1997). *Software engineering: a practitioner's approach*, 4th edition. New York: McGraw-Hill.
- Pressman, R. (2001). "Adaptable process model: software engineering checklists." Accessed at: <http://www.rspa.com/checklists/index.html>
- Wheeler, D., Brykczynski, B., Meeson, R.N., Jr. (1996). *Software inspection: an industry best practice*. Los Alamitos, Calif.: IEEE Computer Society Press.

Biography

Laura Faulkner is Research Engineer/Scientist Associate for the Signal and Information Sciences Laboratory, Applied Research Laboratories, The University of Texas at Austin. SISL designs and delivers experimental software and performs advanced research for a wide-range of defense and other applications. Ms. Faulkner implements and guides the development, test and human factors processes, and assists in design, evaluation and testing. Through her daily work and research she has been developing and refining structured methods for evaluation and testing for several years. Ms. Faulkner is a Ph.D. candidate and has completed a Master's in Psychology, both with an emphasis on human-computer interaction, is completing Process Management Certification, and has earned undergraduate degrees in Anthropology and Sociology. She has also served as the Chair of the Software Quality Institute Advisory Board of The University of Texas at Austin, on the international Human Systems board of the Interservice/Industry Technology, Simulation, and Education Conference, and has an invited instructor at multiple venues on software quality, evaluation, and team process.

Laura Faulkner, Applied Research Laboratories, The University of Texas, P.O. Box 8029, Austin, Texas 78713-8029; telephone 512-835-3328; fax 512-835-3100; laura@arlut.utexas.edu