Managing Peer Reviews Using DataDrill and Code Collaborator

Summary

Introduction

Peer reviews, or inspections of software source code, can significantly raise the quality of a software product by removing defects at an early stage. In this paper, Distributive Management's DataDrill is used to plan, monitor and control the peer review process while Smart Bear's Code Collaborator is used to perform software peer reviews. When used together, DataDrill and Code Collaborator provide an integrated solution for software peer reviews that allows an organization to quickly lower defects, raise software quality and increase management visibility into the software development process.

Business Value

A combined DataDrill and Code Collaborator solution offers a number of key features that are not present in either internally-developed, open source or other commercial solutions:

- Managers can establish planned expectations and targets for peer reviews and then monitor progress toward goals
- Up-to-date progress information is delivered to managers regarding peer review findings and issues
- Built-in rule analysis alerts managers to common mistakes and potential process errors
- Statistical analysis of peer review progress checks for unstable or erratic process implementation
- Ready-to-use best practice indicators for peer reviews
- Pre-built integration requires no technical support

By providing the features above, the DataDrill - Code Collaborator solution empowers an organization to address both management and technical aspects of peer reviews. This focused solution helps your organization achieve its intended business goal for peer reviews: to deliver better software products. Other direct aspects of business value for this solution include:

- Reduce the number of software defects entering unit, systems and integration test activities
- Identify issues with requirement and design documents as early as practical
- Raise the level of software quality for the end user
- Increase management visibility into the software development process

The benefits outlined above are quickly achieved, yielding a short period to realize ROI and to begin developing or expanding a competitive advantage in the delivery of software and software-intensive products. In addition, you can focus your business on true revenue generation, rather than spending valuable resources (and time) defining technical integrations or defining detailed peer review indicators.

Background

A peer review is a desk audit of software source code by developers, other than the author(s), who check the code to identify issues and potential issues. Peer reviews can encompass both the software under review and the requirements and standards being satisfied. For best results, each review should have a small subset of code, perhaps around 400 lines, which are inspected by other members of the team. Each participant in a peer review is given a role, with specific actions and responsibilities to be performed.

During the review, participants read and inspect the software selection and record any issues or potential issues, as well as the effort expended. The author of the software resolves the issues by modifying the software, changing a document, or, when appropriate, by rejecting the issue.

Code Collaborator is a peer code review tool that lets developers easily review code online without extensive meetings, miles of code print-outs, or convoluted email threads. The software automatically gathers changes from version control and packages them for easy distribution to reviewers. Developers can then review and chat directly on the code itself, with Code Collaborator tracking changes and correlating them with the appropriate section of code, even when line numbers change.

More information on peer reviews and Code Collaborator can be found on the Smart Bear web site at www.CodeCollab.com or www.SmartBear.com.

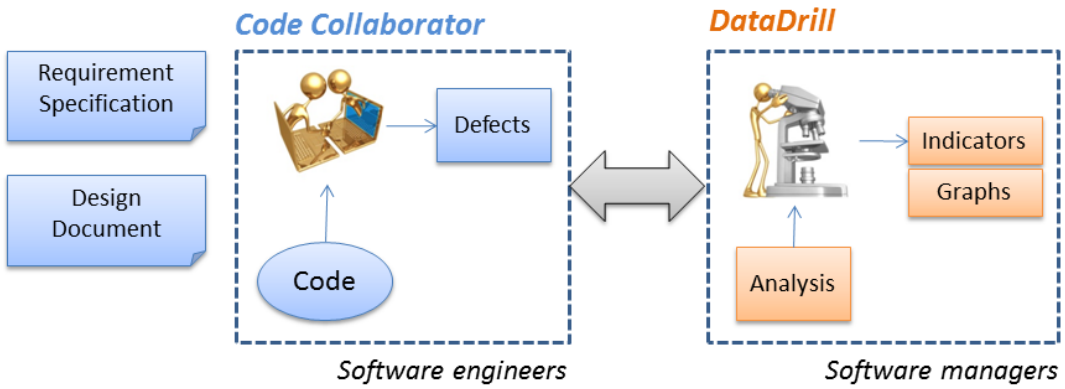
Another essential ingredient to peer review success is the planning and management activities that accompanies the technical activities. DataDrill from Distributive Management is a software management solution that provides managers with the ability to plan, monitor and control critical software engineering processes, including peer reviews. By collecting, analyzing and centralizing all critical software project information, managers are able to use indicators and metrics to quickly spot problem areas and take action, without worrying about the mechanics of data collection and storage.

More information on the DataDrill solution for software indicators is available on the Distributive web site at www.distributive.com.

Managing Peer Reviews with DataDrill

DataDrill and Code Collaborator Operation

A complete peer review solution addresses the needs of both the software engineers who perform the reviews and the managers who plan and control them. The simplified diagram below shows how these functions are related.



Software managers plan the peer review process, which is then performed by the engineers. The data maintained in Code Collaborator is collected, analyzed by DataDrill and then provided to managers in the form of indicators and graphs.

Tailoring for the Peer Review Process

To collect the indicators and measures described in this document, the "default" peer review is augmented with a few pieces of data that must be entered when the peer review is started and when it is completed. This additional data is implemented using the built-in "custom fields" within Code Collaborator, and then extracted by DataDrill during the collection process. In Code Collaborator, we recommend you create the following fields and keep them up to date as a peer review proceeds:

Name	Type	purpose
ProjectId	string 255	A text identifier such as a project, program or sub-system name that will be associated with one or more reviews. This field is used in DataDrill to assign groups of reviews.
PlanStartDate	string 255	Planned start date of the review
ActualStartDate	string 255	Actual start date of the review
PlanEndDate	string 255	Planned finish date of the review

SlocReviewed	string 10	Number of lines of code reviewed during the peer review. Note that this is smaller than the physical or logical size of the module being reviewed.

The fields in the previous table must be added to Code Collaborator to allow management of the peer review process using DataDrill EXPRESS.

Managing Peer Reviews with DataDrill

The DataDrill – Code Collaborator solution provides out of the box capabilities to monitor the progress of peer reviews. In DataDrill, the graphs and indicators are combined into an information need that addresses one area of management interest. For controlling the peer review process, DataDrill provides a peer review information need that addresses the need for managers to:

- Monitor peer review progress
- Manage peer review findings
- Maintain stability of the peer review process
- Enforce peer review rules and best practices

The following subsections describe the DataDrill peer review information need that addresses these management needs. Monitoring Peer Review Progress

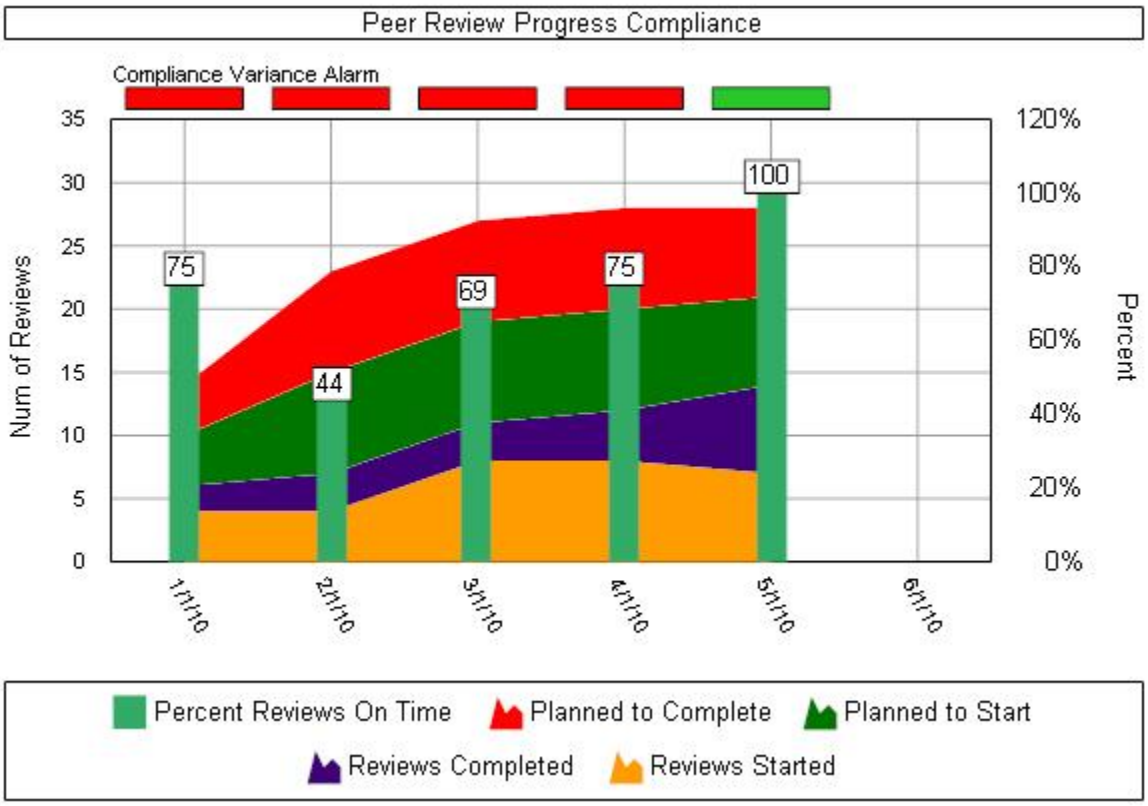
Monitor Peer Review Progress

These graphs are designed to help managers understand the number of reviews in process and the timeliness of completing them. These graphs aid managers in spotting problems in completing reviews before they become critical. Applicable graphs are shown in the following table.

Name	frequency	Description
Peer Reviews By State	schedule	Updated each week, this graph helps managers understand the scope and timeliness of peer reviews planned, in-progress and completed for their project. This run graph shows the number of peer reviews that have been started, opened and closed each period, with the number started being compared to a plan line.
Open Peer Review Aging	schedule	This vertical bar graph helps managers understand how well reviews on their project are being completed once they are opened. This graph displays a count of the number of reviews in pre-selected ages, such as 1 to 5

		days old, 5 to 10 days, 10 to 15 days, 15 to 20 days and then more than 20 days old.
Peer Review Progress Compliance	schedule	This graph allows managers to see how well the project is complying with planned start and end dates for their reviews. This run graph presents the percent of reviews that were started and finished before or on schedule, by comparing the planned start date to the actual start date, and the planned end date to the actual end date.

The following figure is an example of the graphing provided by DataDrill. This graph provides three sets of data of interest to managers: on the left axis and in the stacked area series are the number of reviews planned as well as the number started and completed before their planned dates; the right side axis shows the percent of peer reviews that were started and completed on time; the alarm bar at the top of the graph provides a color status of the number of rules completed on time.

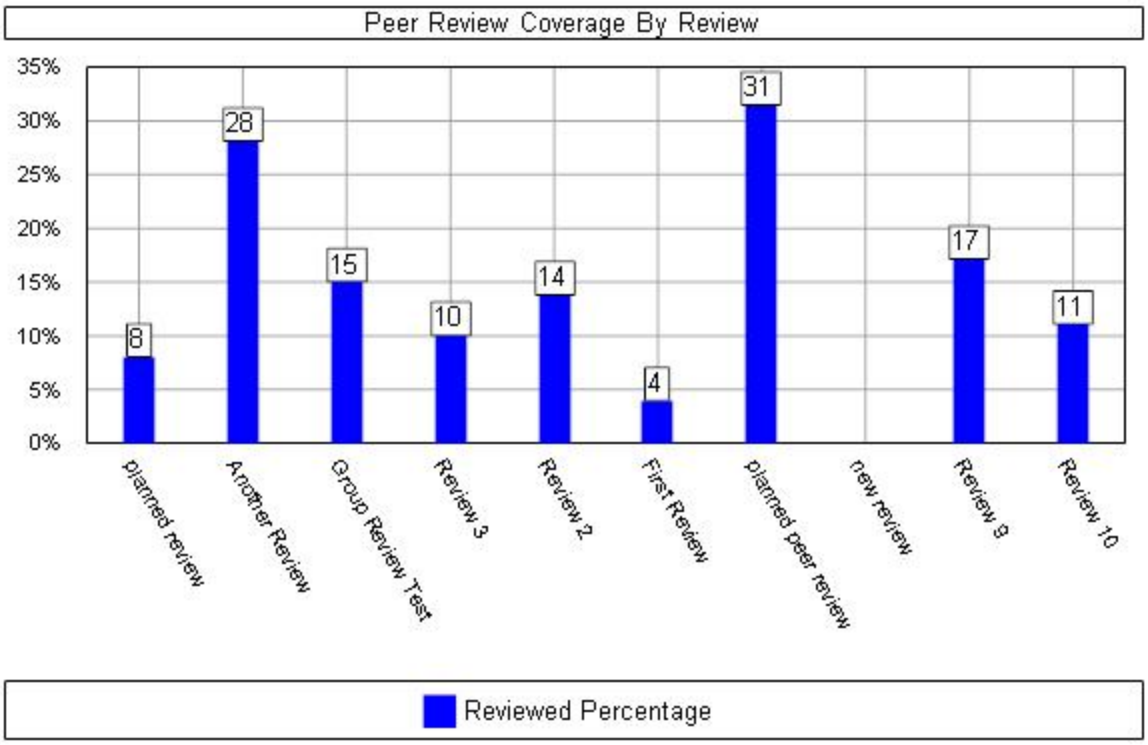


Manage Peer Review Findings

This set of graphs provides insight into the findings associated with each peer review.

Name	Frequency	Description
Peer Review Defects	event	This graph allows managers to review the results of each review as expressed by the number of defects found. This run graph shows the number of defects, the hours, the number of sloc (software(?) lines of code) reviewed and the effective defects per sloc.
Peer Review Coverage Total	schedule	This graph allows managers to understand the amount of code that has been reviewed, as a percent of the total lines of code in the software modules. This graph provides an indication of how many defects are potentially left in the software, which may be found in downstream activities such as unit or systems test.
Peer Review Coverage By Review	event	This graph allows managers to ???verb?? amount of code reviewed as a percent of the total lines of code in the files associated with the review.
Peer Review Effort	event	This graph allows managers to quickly see the amount of effort being spent on peer reviews. This run graph contains the amount of effort expended in each review, as well as a running total of effort.
Peer Review Code Changed	event	This graph displays the lines of code changed, added, modified and deleted as well as total lines of code for all modules inspected.

The following figure, showing a graph of Peer Review Coverage By Review, shows managers how much code (as a percent of the total lines of code) has actually been peer reviewed. Coverage information provides managers with an understanding for the degree of confidence to assume when assessing whether software is ready to proceed into down-stream activities. Software with a low coverage percent is likely to have a higher number of defects and require more testing effort than code with a higher coverage percent.



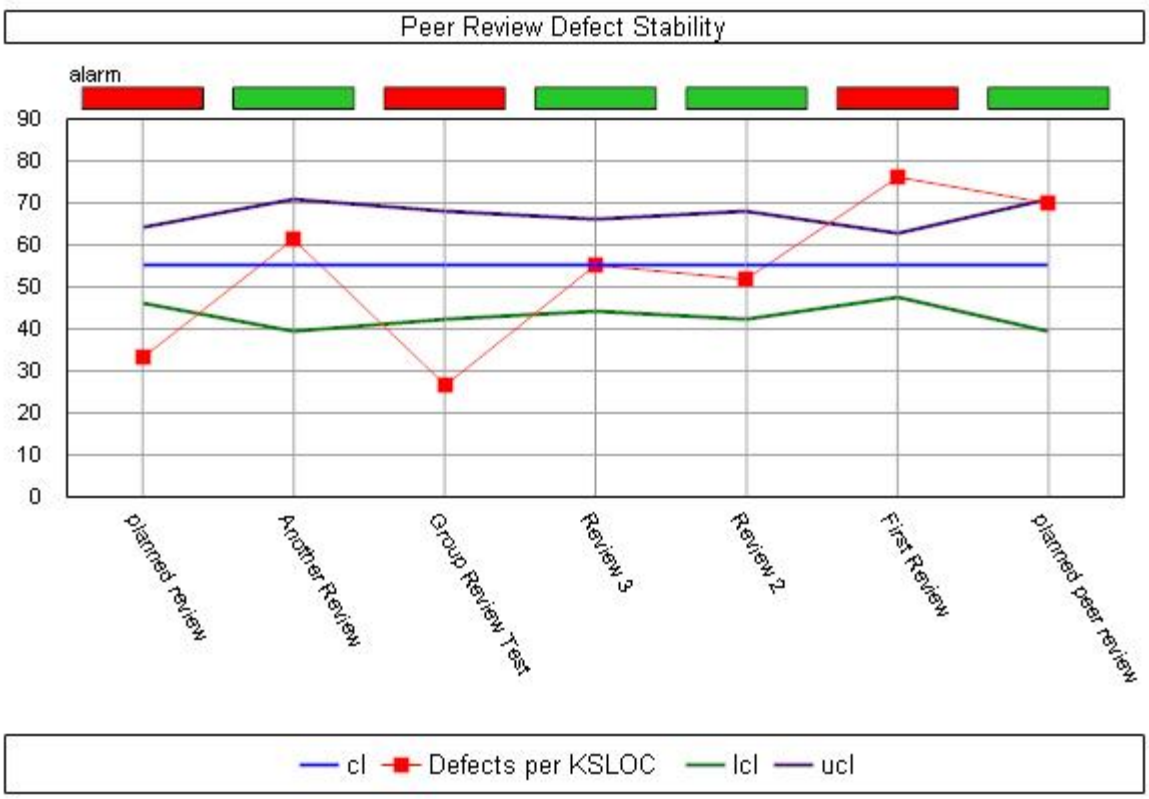
Notice in the graph above that the bottom axis contains actual review names instead of dates. This type of graph in DataDrill, called an event-based graph, is useful when examining raw data and conducting (??) analysis for peer reviews, since they do not occur on a strict periodic basis.

Stability of the Peer Review Process

The peer review process must be periodically monitored to ensure that the process does not appear to be out of control. The idea is that if the peer review process is unstable, then we do not want to use the resulting defect and effort data for decision-making. The determination of what is considered "out of control" is performed using a statistical technique called a control chart using the "U chart" calculation.

Name	frequency	Description
Peer Review Effort Stability	Event	This control chart provides a U chart of the effort expended in each closed/completed review, where the effort is normalized to the amount of sloc reviewed. This run graph also provides alarm series to visually indicate stability.
Peer Review Defect Stability	Event	This control chart provides a U chart of the defects found in each closed/completed review, where the number of defects is normalized to the amount of sloc reviewed. This run graph also provides alarm series to visually indicate stability.

The following graph shows an analysis of the defects found per review as a function of the number of lines reviewed. This graph displays series called "UCL" (for upper control limit) and "LCL" (for lower control limit), which are calculated based on an analysis of the average number of defects found per KSLOC for all reviews. When the "Defects per KSLOC" exceeds the UCL, it indicates that the peer review process may not be being performed correctly and may be what is termed "unstable". An unstable process means that when the results of all peer reviews are compared to themselves, it appears (*or – becomes clear ?) that at least one of the peer reviews generated significantly too many or too few defects.



Peer Review Rules and Best Practices

This set of graphs provides insight into the findings associated with each peer review.

Name	Frequency	Description
Peer Review Rules	Schedule	This table of data, or snapshot grid, contains a set of rules and a count of the number of reviews that violate each rule. The rules include the following: • Reviews that are not started within **5** days of planned start • Reviews that are not finished within **5** days of planned end • Reviews that are open for more than **14** days • Reviews with 0 files that contain defects • Reviews with 0 SLOC that contain defects • Closed reviews with more than **5000** lines of code • Closed reviews with more than **250** defects • Closed review with less than **100** SLOC

		• Reviewer active in more than **3** reviews at the same time • Average lines of code changed per defect exceeds **250** lines Numbers in bold (above) are parameters that can be adjusted by editing the file Accessories\Rules\CCollab.xml
Peer Review Shortcut Table	Schedule	This table contains the title and URL/hyperlink from DataDrill to the project's peer review page in Code Collaborator and date completed.
Peer Review Details Table	Schedule	This table contains the title and URL/hyperlink from DataDrill to the project's peer review page in Code Collaborator and shows: SLOC Total, Defect Total, Comments, Major Defects, Minor Defects, SLOC Reviewed, Review duration and Number of Reviewers.

An example of the Peer Review Rule Grid is shown below.

Peer Review Rule Grid		
▼	1	Reviews have not started within 3 days of planned start
		Review 10
▼	1	Reviews have been open for more than 10 days
		new review
►	0	Reviews with 0 SLOC which contain defects
▼	1	Closed reviews with more than 250 defects
		First Review
▼	2	Closed reviews with more than 5000 lines of code
		planned review
		First Review

Note: The rules and trigger values in the grid can be tailored in the field.

Summary

When used together, DataDrill and Code Collaborator provide an integrated solution for conducting software peer reviews that allows an organization to quickly lower defects, raise software quality and increase management visibility into the software development process. By controlling defects during the coding phase of a software project, organizations can remove more defects earlier, when it is cheaper and easier to find them.

The process presented here captures industry best practices and makes them easily deployable in a short amount of time. Further, the joint DataDrill / Code Collaborator solution provides both the technical and management aspects of peer reviews with a minimum of technical or infrastructure requirements.

Contact

Please contact us to learn more!

Distributive Management

107 Olde Greenwich Drive, Suite 101

Fredericksburg, Virginia 22408

www.distributive.com

info@distributive.com

800-779-6306

540-891-8885 (fax)