Significance of Peer Reviews in Software Quality Control

Executive Summary

Peer review is one of the best techniques for controlling defects in the software development work products. This article details how peer reviews can be done more methodically and effectively Peer reviews bring down the development cost and improves the final quality of the software. Also some steps have been defined to monitor the maturity of Peer reviews within an organization/team. Some thumb rules are prescribed for the readers to set their initial benchmarking

What is Peer Review?

According to CMM, peer review is a methodical examination of software work products by the producers' peers to identify defects and areas where changes are required. . The specific components that will undergo peer reviews are identified and planned as part of the software project plan.

The purpose of peer reviews is to identify and remove the defects from the software work products early and efficiently. An important corollary effect is to develop a better understanding of the software work products and of defects that might be prevented. Also peer reviews prepares an alternative author for the reviewed components so that, part of the development risks due to possible resource exodus is taken care at the beginning of the development life cycle.

Types of Peer reviews

Three types of peer reviews are done normally in typical software development life cycle and they are briefly described here:

1. Code/Document Walk through: In this process the author and a set of reviewers with different backgrounds gather in a meeting room. The author of a document or software component walkthrough his/her work product to the review participants from a layman's perspective. The participants are free to stop him at any moment and pose questions and take clarifications. The team does not review the code but reviews the functionality, logic and issues pertaining to the ultimate user. The author takes note of the varied views of the participants and makes necessary modifications to his document /software.
2. Peer Reviews with single co-author: In this kind of Peer review a co-author is assigned for each component identified for review. The author gives a kick off to his co-author about the work product and delivers the component/document. The co-author acquires an understanding about the component through a detailed review and testing. The defects are identified with the help of appropriate checklists, and the co-author records them as review remarks. The author reviews the defects and solves them to close the remarks. The reviewer retests/verifies the solution after the rework
3. The third type of review is popularly known as Fagen Inspection. Here a team reviews the component in a structured manner. Following roles are assigned in a fagen inspection:
    - Author who developed the component
    - Inspector : there can be two or more inspectors assigned depending on the document/component software
    - Moderator : who plans the inspection and moderates the meeting
    - Scribe : Who records the errors

The last type of review is more effective but most costly too. The investment required to do a fagen inspection is typically three times that of a walk through or peer review with single co-author. In this paper we focus more on the second type.

Peer Review – Some Dos and Don'ts:

- Assigning resources on a project is an important task for a project leader. As far as possible a cluster of components/modules having the same or related functionality is to be assigned to an author for development. The same set of components to be assigned to single co-author for the purpose of peer reviews. This facilitates them to have a continued relationship for the whole development life cycle of the product. The author and the co author work as a team.
- Peer review teams are to be identified during early stage of development. This can happen the moment components are identified and the knowledge is imparted to the component authors
- Check lists are very important to the success of peer reviews. These checklists guide the peer review process and ensure consistency across the product.
- Entry criteria for Peer review are to be defined and documented. The author is responsible to fulfill the entry criteria before he asks for the review of a component. Some of the components of entry criteria could be
    - A technical unit test should be conducted by the author and the component should be in a working condition
    - The code must be fully documented and all unrelated/commented lines to be cleaned
    - The component should be handed over fully along with UI, Code, reports and the designs Depending on the type of component the definition of entry criteria will vary
- The reviewer should follow a set of guidelines to do a peer review. This can be defined for different work products. For e.g. if it is a requirements document the review rate could be 2 – 3 pages per hour and if it is a software code the rate could be 200 – 250 non commented lines per hour.
- All Peer Reviews should be planned and performed following a predefined process and using instruments like log sheet, check lists and entering the review results in the defect tracking system. The review remarks should be analyzed by the author, resolved, reworked and closed.
- The process data should be collected across the project and organization to analyze the process stability and to make necessary improvements.

Case study on Peer Review Process and Performance

Introduction

The project being discussed under this case is a new product development forming a part of an ERP package. A new module called ***Project Estimation*** was added to the existing version of ERP suite. The size of the Package under review is about 35000 non-commented source code involving 32 components in it.
The development life cycle consists of the following stages:

- Conceptual solution
- Functional Design
- Technical Design(detailed design)
- Coding and unit test
- Integration test
- System test

Peer reviews are done on the work products during the first four stages of development.

The scope of this case study is limited to the analysis of Peer review process on the source code.

Review Process

The review process is according to the Baan Company's internal Software Development Method called BDM (Baan Development Method) and its tailoring version as specified through the QA plan.

The broad guidelines are:

- All components must be peer reviewed
- The review rate should be 200 non commented source code per hour
- The average yield together with Fagen Inspection should be around 60%
- All peer reviews should be formally registered in the Fagen Inspection system (Defect Tracking tool) and remarks / errors to be handled to the satisfaction of the peer
- The reviewer should use the relevant checklists

Review performance and summary results

The review performance was monitored with the following common parameters

- Size of the component as Kloc
- Review rate in non commented source code statements per hour
- Out put as Defects per Kloc
- Review number

As and when the reviews are completed the results are summarized in an excel file. The table below (Page 4)gives the summarized details of all the reviews done on the project.

Control chart technique has been used to monitor the review process and its effectiveness. The review data (results) have been plotted in a control chart with the first few weeks of results (with 10 to 12 data points) and then updated fortnightly with additional results. The analysis was shared with the project team every fortnight during the progress review meetings. This has resulted in the following positive benefits to the team and its performance.

- All the team members received an update about the progress and effectiveness of the work performed
- In case of some components not showing up the desired performance, it was discussed further (for the cause and effects) and a decision was taken to re-review the component

Table 1 - Peer Review Results Summary					
Component Number	Lines of Code	Defects	Meeting time (minutes)	Review rate (lines of code per hour)	Output (defects/K loc.)
1	334	4	230	87	12.0
2	448	6	270	100	13.4
3	293	2	50	352	6.8
4	520	4	120	260	7.7
5	470	3	180	157	6.4
6	1110	11	540	123	9.9
7	1103	9	360	184	8.2
8	3414	18	960	213	5.3
9	2118	6	480	265	2.8
10	800	3	180	267	3.8
11	719	9	360	120	12.5
12	200	0	60	200	0.0
13	2400	31	480	300	12.9
14	1177	8	540	131	6.8
15	1605	16	360	268	10.0
16	3900	35	1500	156	9.0
17	293	6	135	130	20.5
18	282	8	105	161	28.4
19	118	1	105	67	8.5
20	235	6	105	134	25.5
21	2682	11	720	224	4.1
22	891	4	240	223	4.5
23	1698	17	360	283	10.0
24	1816	18	290	376	9.9
25	960	16	360	160	16.7
26	847	8	360	141	9.4
27	399	14	240	100	35.1
28	300	2	120	150	6.7
29	206	2	120	103	9.7
30	2900	22	1140	153	7.6
31	125	1	90	83	8.0
32	150	5	120	75	33.3
33	952	8	420	136	8.4
Total	**35465**	**314**	**11700**	**182 (average)**	**8.9 (Average)**

Review Maturity – Introduction

The data was mapped and analyzed in the form of the following two control charts

1. Review rates
2. Yield as defects per Kilo lines of non commented code

Analysis – Chart I – Review Rates

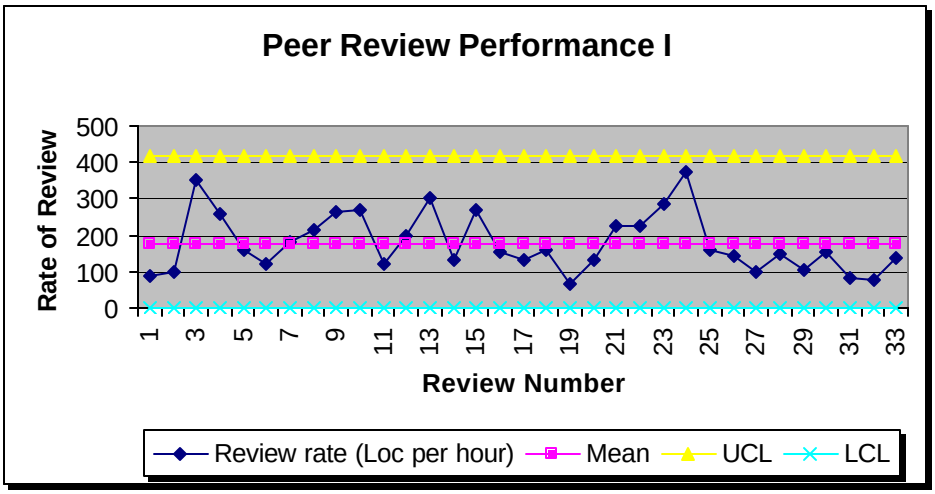
In chart I following four results are plotted

- The actual data value (Rate of review in non commented source code statements per hour)
- The mean value of all the review rates (constant – 182 lines per hour)
- The Upper control limit (UCL). This is calculated as follows:

  UCL = Mean + 3* SD
  = 420
- The Lower control limit(LCL) This is calculated as follows:

  LCL = (Mean – 3*SD) or Zero whichever is lower
  = 0

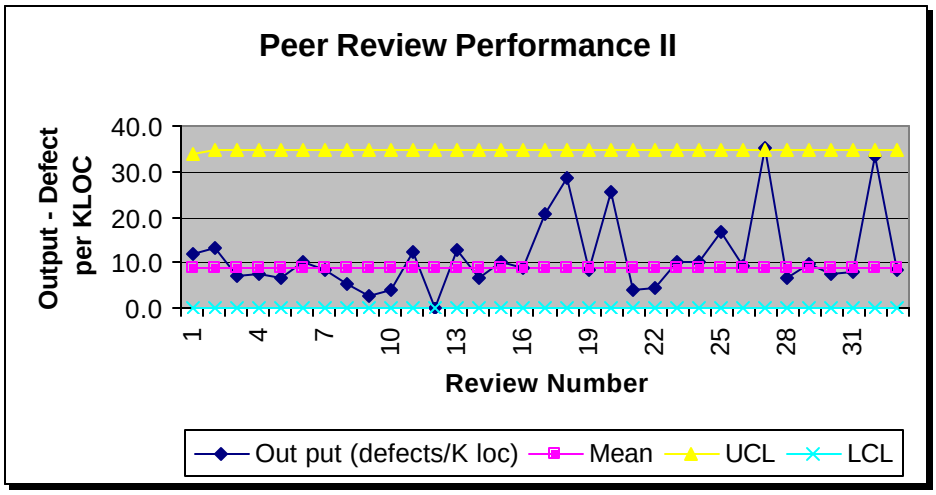
It is observed that the process is under control except between data points 25 and 34 where nine consecutive data points are below the centerline. However this variation is more on the positive side as lesser review rate is good for the product. From the team perspective it is important to maintain this performance. The awareness of this performance review thereby makes the team members more confident about their reviews and helps set personal targets to ensure that they follow the guidelines.



Analysis Chart II – Yield

Chart II represents the yield data in the form of defects per Kilo lines of non commented source code in the same lines as that of chart I. The process in general is under control. This performance review reflects the effectiveness of the review and a good relationship is seen between chart II

and I. In most instances, if the review rate is below the average line (chart I) the output in terms of defects per k loc. (chart II) is above the average line. One review has resulted in a zero yield and two reviews are showing assignable (special) cause variation. These cases are analyzed for the possible reasons and necessary corrective actions initiated.

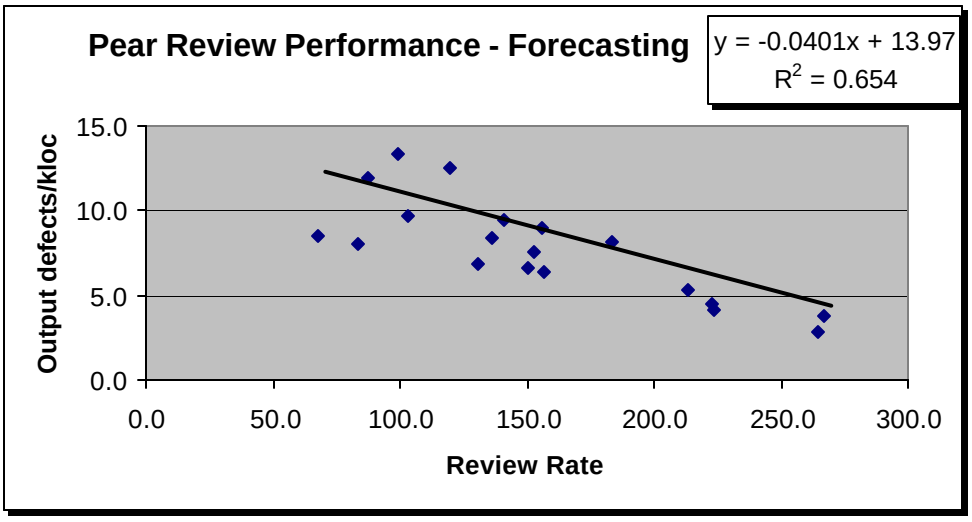


Review results forecasting

The extreme data points from the population have been removed and the results have been plotted in a scatter plot and tested for the correlation. The outcome was more positive and helped as a guide to perform future reviews. A correlation co-efficient of 0.654 (R^2 value of > 0.5 and <0.7 indicates that there is an adequate correlation for many purposes) is more reliable for usage in other similar applications. The simulated data values are presented in the table below, which could be used as a tool for planning.

Table II:

Review rate	Output
50	12
75	11
100	10
125	9
150	8
175	7
200	6
225	5
250	4
275	3
300	2
325	1
340	0.4



Peer Reviews - cost benefit analysis

The economics of software quality largely depends on the cost of error detection, prevention and removal. While it is important to produce quality products, each test activity costs resources and takes time. The key to managing this issue is to recognize that a quality product is put into the test system.

The statistics on the number of defects removed per phase and efforts to find the defects are summarized in the below table:

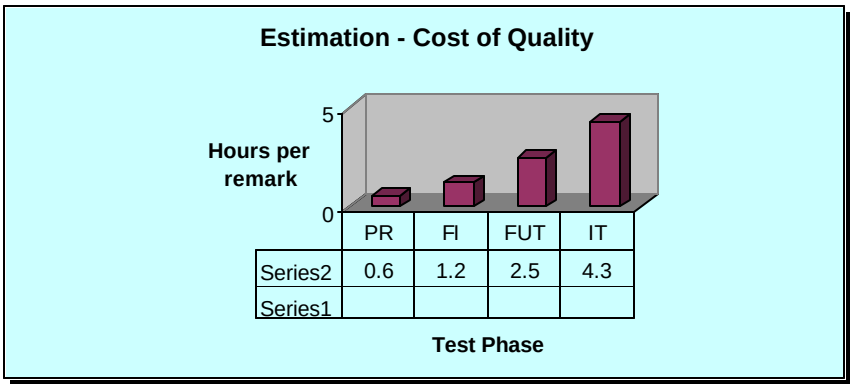
Test Phase	Peer Review	Fagen Inspection	Functional Unit Test	Integration test	Total / Average
Remarks	**1090**	**296**	**223**	**135**	**1744**
Percentage of remarks	**63**	**17**	**13**	**8**	**100**
Hours invested	**603**	**361**	**556**	**575**	**2095**
Time per remark	**0.6**	**1.2**	**2.5**	**4.3**	**1.2**

Following key observations could be made while analyzing the data

- Assuming another 8% of the defects are left to the subsequent test phases and to the field, 58% of the defects could be removed if the peer reviews are done to 100 % of the code with a standardized process

- It is seen that the time to find a defect from peer review to Integration test has increased seven folds.

The chart below on the same data illustrates the exponential increase of efforts (cost) as we progress in to the further phases of testing the software.



Conclusion:

The analysis of the above case leads us to the following conclusions:

1. Peer Review is one of the critical process in any development lifecycle.
2. Peer review process needs to be defined and managed as a part of the project plan.
3. Peer review Process also helps to manage the project risks.
4. Peer reviews bring down the cost of quality in the form of reducing the test effort and duration. Time to market and predictability improves.
5. Peer review process can be statically controlled to monitor the effectiveness and to take necessary corrective / preventive actions.

P Seenivasan,
Release Manager,
Baan Infosystems India Limited,
Hyderabad – 500 033
India

e-mail: pseeniva@baan.com