Software Engineering Practitioners' Research Methods Chapter 20 Answers

Problem:

Explain the difference between an *error* and a *defect.*

Answer:

4633-15-1P SA: 9420

SR: 6376

• An **error** is a quality problem that is discovered by software engineers (or others) **before** the software is released to the end user or to another framework activity in the software process, and

• A **defect** is a quality problem that is discovered **after** the software has been released to end users or to another framework activity in the software process.

Problem:

Why can’ t we just wait until testing to find and correct all software errors?

Answer:

We can’ t just wait until testing to find and correct all software errors because of several reasons. They are

• Detecting all the errors at the end, in the testing phase will increase the effort.

• If we find errors early in the process, it is less expensive to correct them.

• Errors have a way of amplifying as the process proceeds. So a relatively minor error left untreated early in the process can be amplified into a major set of errors later in the project.

• If we wait till testing without finding and correcting the errors, they become major errors.

• Finding and correcting errors partially in the early stages will save time.

• By detecting the errors in the early stages, the errors cannot be propagated to the next step.

• By detecting and removing a large percentage of errors in the early stages will reduce the cost of subsequent activities.

Problem:

Assume that 10 errors have been introduced in the requirements model and that each error will be amplified by a factor of 2:1 into design and an addition 20 design errors are introduced and then amplified 1.5:1 into code where an additional 30 errors are introduced. Assume further that all unit testing will find 30 percent of all errors, integration will find 30 percent of the remaining errors, and validation tests will find 50 percent of the remaining errors. No reviews are conducted. How many errors will be released to the field.

Answer:

Number of errors introduced in requirements analysis = 10

As each error will be amplified by a factor of 2:1 into design, number of errors carried into design analysis = 05

Total number of errors in the beginning of design analysis=15 errors

(10 preliminary errors, 5 amplified errors).

Additional errors introduced in design analysis = 20

So Total Number of errors in design analysis = 35

As each error will be amplified by a factor of 1.5:1 into code, number of errors carried into code = 24 (35/1.5)

Additional errors introduced in code = 30

So Total Number of errors in code = 89 (35 + 24 + 30).

Hence the total number of errors at the beginning of the testing phase is 89 errors.

Percent of errors found during unit testing = 30%

Number of errors found during unit testing= (89*30)/100 =27

Nickname: mike6606
Age: 8 years 10 months
Fans: 0
Followers: 5
+ Plus follow

< March 2023 >

day	One	Two	Three	Four	Five	Six
26	27	28	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Search

Look for it
Google search

Frequently used links

My essays
My comment
My involvement
Latest comments
My labels

Essay archives

January 2021 (1)
January 2018 (3)
January 2017 (8)

Read the leaderboard

1. Software Engineering Practitioners' Research Methods Chapter 13 Answers (1578)
2. Software Engineering Practitioners' Research Methods Chapter 1471 Answers (<>)
3. Software Engineering Research Methods for Practitioners Chapter 1290 Answers (<>)
4. Software Engineering Practitioners' Research Methods Chapter 34 Answers (1282)
5. Software Engineering Practitioners' Research Methods Chapter 1264 Answers (<>)

Recommended leaderboards

1. Software Engineering Practitioners' Research Methods Chapter 1 Answers (<>)

Number of errors found in integration testing=(62*30)/100=19 errors

Number of uncovered errors in integration testing=(62-19)=43 errors

Percent of errors found during validation testing = 50%

Number of errors found in validation testing-(43*50)/100=22 errors

Thus 22 errors are released to the field.

Problem:

Reconsider the situation described in Problem 20.3, but now assume that requirements, design, and code reviews are conducted and are 60 percent effective in uncovering all errors at that step. How many errors will be released to the field?

Answer:

10 errors have been observed in the requirements model. 60% of these errors, i.e. 6 errors are uncovered in review meetings. Thus 4 errors are amplified by a factor of 2:1. Each of the two errors would amplify 1 more error. Therefore, the 4 errors amplify 2 extra errors resulting in a total of 6 errors (4 + 2).

These 6 errors are observed into the design phase also which has 20 newly generated errors also resulting in a total c 26 errors. 60% of these errors, i.e. 16 errors are uncovered in review meetings leaving 10 errors uncovered in this phase.

In coding, these 10 errors from the design phase get amplified by a factor of 1.5:1 which accounts for nearly 7 (10/1.5 amplified errors. Also, there are 30 newly introduced errors. Therefore, the total number of errors is 47 (10 + 7 + 30). 60% of these errors, i.e. 28 errors are uncovered in review meetings leaving 19 errors uncovered in this phase.

Now, in testing, 30% of these errors i.e. 6 are uncovered in unit testing leaving 13 errors (19 - 6). 30% of the 13 errors i.e. 4 errors are uncovered in integration testing. Now the number of errors left is 9 (13-4).

In the validation testing phase, 50% of the 9 errors are uncovered. Thus 5 errors are released to the field.

Problem:

Reconsider the situation described in Problems 20.3 and 20.4. If each of the errors released to the field costs $4,800 t find and correct and each error found in review costs $240 to find and correct, how much money is saved by conducting reviews?

Answer:

Without reviews, 22 errors are released to the field. Thus the total cost for find and correct all errors are 22*4800 = $105600.

Total errors uncovered in reviews are 6+16+28=50. The cost to find and correct these errors is 50*240 = $12000. 5 errors are released to the field which would cost 5 * 4800 = $24000. Thus the total cost to find and correct all errors i 12000+24000=36000.

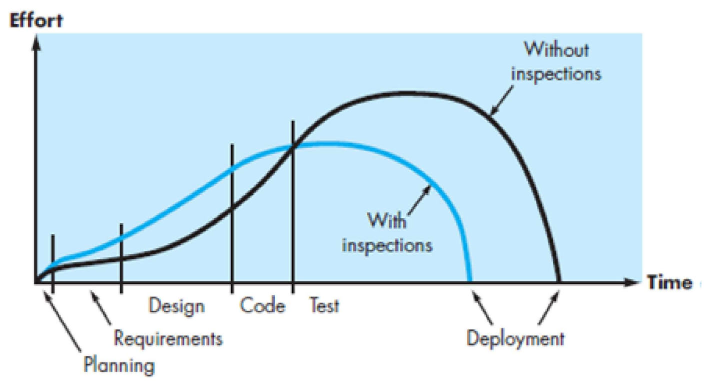
Therefore by conducting reviews, the amount of money saved is 105600 – 36000 = $69600.

Problem:

Describe the meaning of Figure 20.4 in your own words.

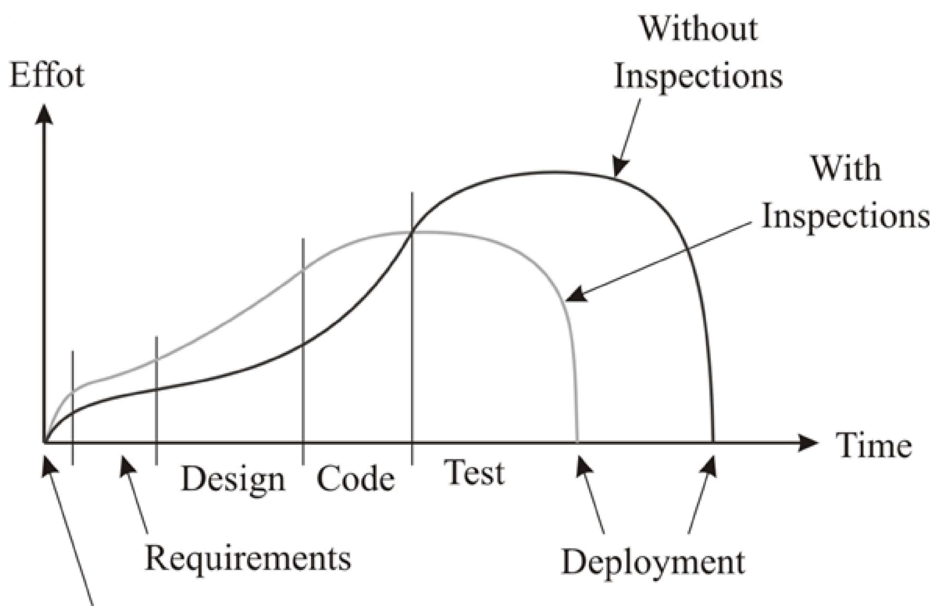
FIGURE 20.4 Effort expended with and without reviews

Source: Adapted from [Fag86].



Answer:

4633-15-6P SA: 4475



• In this figure, time is taken on X-axis and effort on Y-axis.

• With inspections/reviews, the effort increases gradually in the early stages and decreases in the testing phase.

• But without any reviews, the effort in the early stages will be less, but it increases in the testing phase.

• Also in the case of development without reviews, deployment time increases. That means the deployment of the software becomes late, if the reviews are not considered. With reviews the deployment time becomes less.

Problem:

Which of the reference model characteristics do you think has the strongest bearing on review formality? Explain why

Answer:

4633-15-7P SA: 4475

SR: 6376

Reference model characteristics:

• Planning & preparation

• Meeting structure

• Correction & verification

• Roles individuals play

Of all reference model characteristics, **Planning and preparation** has the strongest bearing on the review formality, because, it helps to define the level of review formality. A set of specific tasks would be conducted based on an agenda that was developed before the review occurred. Without proper planning and preparation one cannot decide on what issues they need to work. And if there is no proper planning, some issues to be reviewed may remain uncovered.

Problem:

Can you think of a few instances in which a desk check might create problems rather than provide benefits?

Answer:

Desk checking is a method to verify the portions of the code for correctness. The main advantage is that the programmer who knows the programming language and code very well is equipped to read and understand his or her own code. It is carried out by individual programmers and analysts who were not involved in the software produc creation.

Following are the few instances in which a desk check might create problems rather than provide benefits.

• As the person who has no involvement in the programming will do the desk checking, he may not have complete idea of the logic that the programmer used. Problem comes when he has mistaken the logic and treats it as an error.

• If the person who performs the desk check does not have proper knowledge, then the process will not have any use In addition to that, it creates additional problems of making some unnecessary corrections.

Problem:

A formal technical review is effective only if everyone has prepared in advance. How do you recognize a review participant who has not prepared? What do you do if you' re the review leader?

Answer:

4633-15-9P SA: 4478

SR: 6376

Formal technical review:

A formal technical review (FTR) is a software quality control activity performed by software engineers. Each FTR is conducted as a meeting and will be successful only if it is properly planned, controlled and attended.

Each reviewer is expected to spend between one and two hours reviewing the product, making notes, and otherwise becoming familiar with the work. During the FTR, reviewers raise issues based on their advance preparation. So, depending on the issues being raised and questions being asked, it can be recognized whether a review participant has prepared or not. The participant, who did not prepare in advance, cannot raise any questions. A participant who didn' t participate in the discussion can be considered that he has not prepared.

In such cases, being a review leader he can motivate the participants by making them understand the use of FTR and the importance of the advance preparation for FTR.

Problem:

Considering all of the review guidelines presented in Section 20.6.3, which do you think is most important and why?

Answer:

4633-15-10P SA: 9420

SR: 6376

Of all the review guidelines, the guideline, "Develop a checklist for each product that is likely to be reviewed" can be the most important one.

A checklist helps the review leader to structure the FTR (formal technical review) meeting and helps each reviewer to focus on important issues. The main purpose of conducting FTR is to find errors and uncover issues that would have negative impact on the software to be deployed. This guideline of developing a checklist plays a major role in achieving the goal of FTR. Hence we can consider this guideline as the most important one.

Solution: Chapter 20: REVIEW TECHNIQUES

20.1 Errors are quality problems discovered before releasing a work product to end-users or another framework activity, defects are quality problems discovered after releasing a work product to end-users or another framework activity

20.2 Early detection of defects will reduce development costs and reduce the amount of rework required to correct defects. Testing occur to late in the development process to correct design defects and requirements conformance problems.

20.3 10 requirements errors amplified 2:1 to become 20 errors in design

20 new design errors introduced 20 + 20 = 40 and amplified 1:1.5 to become 60 code errors

30 new errors introduced during coding 30 + 60 = 90 at the start of testing

30% of 90 errors uncovered during unit testing leaving 90 – 0.3 * 90 = 63 errors

30% of 63 errors uncovered during integration testing leaving 63 – 0.3 * 63 = 45 errors

50% of 45 errors uncovered during validation testing leaving 45 – 0.5 * 45 = 23 errors

20.4 10 requirements errors with 60% caught during reviews reducing the error count to 4 which are amplified 2:1 to become 8 errors in design

30 new errors introduced during coding 30 + 26 = 56 with 60% caught during reviews leaving 34 errors at the start of testing

30% of 34 errors uncovered during unit testing leaving 34 – 0.3 * 34 = 24 errors

30% of 24 errors uncovered during integration testing leaving 24 – 0.3 * 24 = 17 errors

50% of 17 errors uncovered during validation testing leaving 17 – 0.5 * 17 = 9 errors

20.5 The errors released to the field in 15.3 cost 23 * $4800 = $110400 to fix in 15.3

The errors released into the field in 13.4 cost 8 * $4800 = $43200 to fix and cost (6 + 11+ 22) * $240 = $9360 to detect and fix in 15.4

20.6 The earlier an error is detected the less effort is required to correct it. Projects without reviews generally take longer to complete than projects that use reviews.

20.7 The amount of planning and preparation required, when work products get larger the effort required to review them becomes greater and the likelihood and coordination and reporting becomes more time consuming as well.

20.8 It’ s generally a bad idea to review incomplete products. Exclusive use of desk checking may encourage premature examination of work products and encourage the reviewers to focusing on level implementation details rather than over all system architecture. This is bad for safety critical systems – it may not be bad for WebApp that are easily developed as separable components.

20.9 Typically, an unprepared reviewer will be reading the product materials while everyone else is conducting the review; will be especially quiet throughout the review; will have made no annotation on product materials; will make only very general comments; will focus solely on one part (the part he/she read) of the product materials. As a review leader, you should ask if the person has had time to prepare. If most reviewers have not prepared, the review should be cancelled and rescheduled.

20.10 “Review the product not the producer” . It is important to recognize that there are many ways to meets the customer’ s needs. The important part of software development is creating work products that are high quality. When developers fear that reviewers are looking to criticize their abilities developers are more inclined hide product weaknesses and less inclined to work collaboratively to identify and resolve work product deficiencies.

好文要顶 关注我 收藏该文

mike6606
粉丝 - 0 关注 - 5
0 0
+加关注

« 上一篇： 软件工程 实践者的研究方法 第19章答案
» 下一篇： 软件工程 实践者的研究方法 第22章答案

posted @ 2021-01-21 15:34 mike6606 阅读(1268) 评论(0) 编辑 收藏 举报

刷新评论 刷新页面 返回顶

登录后才能查看或发表评论，立即 登录 或者 逛逛 博客园首页

【阿里云】2核2G云服务器低至99元/年，百款云产品优惠享不停

编辑推荐：
· 现代图片性能优化 - 图片资源的容错及可访问性处理
· 浅谈：服务架构进化论
· 我又和 redis 超时杠上了
· 巧用 CSS 变量，制作高级感拉满的网格动画
· 记录一次锁的优化

阅读排行：
· 使用 Vue 3 时应避免的 10 个错误
· ASP.NET Core Web API 接口限流
· 【故障公告】cc攻击又来了，雪上加霜的三月
· 现代图片性能优化及体验优化指南 - 图片资源的容错及可访问性处理
· 如何在 Net6.0 中对 WebAPI 进行 JWT 认证和授权

Copyright © 2023 mike6606
Powered by .NET 7.0 on Kubernetes