


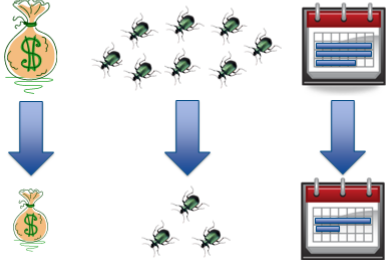
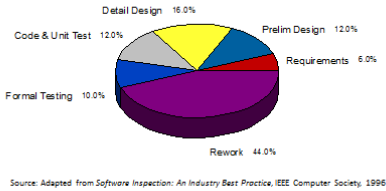
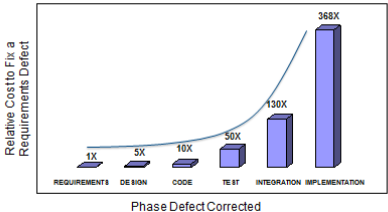
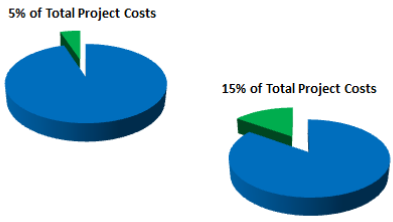


1	 <h2 style="text-align: center;">Software Quality Reviews</h2> <p style="text-align: center;">Costs, Benefits, Critical Success Factors</p>	
2	 <h2 style="text-align: center;">Lecture Topics</h2> <div style="display: flex; align-items: center;">  <div> <ul style="list-style-type: none"> • Discuss the benefits of doing software quality reviews • Discuss the costs associated with doing software quality reviews • Discuss critical success factors for doing software quality reviews </div> </div>	<p>In this lecture, we will discuss the reasons for doing software quality reviews, how quality reviews add-value and improve quality, the costs and benefits of doing quality reviews, and critical factors for ensuring that reviews will be successful.</p>
3	<h2 style="text-align: center;">Why Do Quality Reviews?</h2> <div style="text-align: center;">  </div>	<p>Before I even discuss what software quality reviews are, I want to talk about why they should be done. One important reason is that they result in higher quality software products. By higher quality products I mean products that are delivered with fewer defects. When I help clients improve their software engineering and management processes, and we talk about quality reviews resulting in better quality products...that's a no brainer...and I get no arguments from them.</p> <p>However, when I ask them if they're doing software quality reviews things get kind of interesting. Sometimes they say no, and sometimes they say yes...but the most frequent answers I get go something like this: "we try to do them, but we don't always have time", or, "we used to do them, but they took too much time, so we dropped them."</p> <p>Then, when I tell them that quality reviews can both reduce cost and shorten the project schedule, they look at me a bit strangely...like...how can that be? And that makes it pretty clear that they don't understand what I call the "physics" of software quality reviews. Let me explain.</p> <p>Done effectively, quality reviews can reduce the total</p>

		<p>project cost. By that, I mean less effort...less person-hours...are required. This happens as the result of finding and eliminating defects earlier in the project life cycle, where defect detection is much cheaper compared to later in the life cycle...and this, in turn, reduces the amount of total project rework. Reduced total effort also results in shortened schedules.</p> <p>So...in a nutshell...this is what I like to call the "physics" of software quality reviews. We spend some additional time doing quality reviews, but this investment is more than paid back, resulting in less effort, shortened schedule, and, of course, a higher quality product.</p> <p>The key to achieving these benefits is that quality reviews must be "effective"...and I'll describe in later lectures what I mean by "effective". I'll also discuss in more detail in later lectures how effort and rework are actually reduced.</p> <p>For now, let's look at some specific examples of benefits that organizations have realized from quality reviews.</p>
4	<p>Sample Quality Review Benefits</p>  <p>[1] Tom Gilb and Gloria Graham, <i>Software Inspection</i>, McGraw-Hill, 1993 [2] Steve McConnell, <i>Rapid Development</i>, Microsoft Press, 1996 [3] <i>Software Inspection: An Industry Best Practice</i>, IEEE Computer Society, 1996 [4] Roger Pressman and Bruce Maxim, <i>Software Engineering: A Practitioner's Approach</i>, McGraw-Hill, 2004</p>	<p>Let's start with schedule. Effective use of quality reviews has shortened project schedules by 10 to 30 percent. That's pretty significant. For a project that would otherwise take, say 12 months, the schedule might be shortened to a little over 8 months...a savings of more than 3 months.</p> <p>Software quality reviews are actually more effective than traditional software testing at detecting defects. Traditional software testing defect detection rates average out to about 60 percent. Depending on the type of quality review process used...defect detection rates can approach 90 percent.</p> <p>Implemented effectively, software quality reviews help to significantly improve the quality of delivered products...resulting in products delivered with 10 times less latent defects.</p> <p>As an example of cost reduction and increased productivity, and by applying software quality reviews during the requirements phase, some have found that</p>

		<p>a single requirements defect detected and removed as a result of a requirements review can save 30 hours of testing effort. That's also quite significant.</p>
5	<div data-bbox="274 333 756 1293"> <h3>Typical Project Rework Costs</h3>  <p>Source: Adapted from <i>Software Inspection: An Industry Best Practice</i>, IEEE Computer Society, 1996</p> </div>	<p>I mentioned that reduction of rework is one of the positive outcomes of performing successful reviews. So...what is rework anyway?</p> <p>Rework consists of unplanned repeating of tasks that were already performed...because they weren't done correctly the first time or two. Rework is also performing tasks that should have been performed earlier, but weren't...maybe because the project team was trying to cut corners.</p> <p>From an industry standpoint, rework accounts for a significant portion of project cost. There have been numerous studies done to measure this that have found that rework can range from about 40 percent to more than 50 percent of total project costs...that's about half of all costs...and that's very, very significant.</p> <p>Let's take a look at the 44 percent figure on the slide. Suppose we were able to do something that cut the rework percentage in half to say...20 percent. So, right away the project cost is reduced by 20 percent, and the schedule is shortened as well...not necessarily by 20 percent, but any time you reduce effort the schedule can almost always be shortened.</p>
6	<div data-bbox="274 1293 756 1896"> <h3>Defect Detection & Repair Costs</h3>  <p>Adapted from T. Bennett & P. Weinberg, 'Eliminating Embedded Software Defects Prior to Integration Test', <i>Quality Talk</i>, December 2008</p> </div>	<p>There have been numerous studies in industry in which the relative cost to find and fix software defects has been measured as a function of time. All of those studies have found that the cost of finding and fixing defects increases over time in an exponential-like way.</p> <p>This graph shows the results of one such study. In this study they were measuring the relative cost of finding and fixing requirements defects as a function of life cycle phase. Here's how we read it...if it costs 1 monetary unit to find and fix a defect during the requirements phase, it can cost 10 times that amount to find and fix the defect in the coding phase, 50 times that amount to find and fix it in the test phase, and more than 300 times that amount once the product has been implemented.</p>

		<p>That's pretty scary stuff. Other studies have indicated costs of finding and fixing defects between 10 and 200 times the cost relative to the requirements phase. If you fit a curve through the data points it would have a generally exponential shape.</p> <p>So...what causes this generally exponential relationship? One thing that causes it is the extent of rework that is required. The later a defect is detected, the more rework potential there is. A defect found in the requirements phase can be quickly fixed. A defect found in the test phase requires more work...we have to find the root cause of the problem, which might require going back and changing a requirement, then changing a design, then changing code, and then repeating multiple levels of test.</p> <p>You can see the idea...the further downstream you go before you detect a problem, the further upstream you may have to return, and then move downstream again...this is rework.</p>
7	<p>Software Quality Review Costs</p>  <p>5% of Total Project Costs</p> <p>15% of Total Project Costs</p>	<p>Any expenditure of staff effort incurs cost, and performing quality reviews is no exception.</p> <p>Performing software quality reviews can range from 5 percent to 15 percent of total project effort, depending upon the type of review process used and how many reviews need to be performed.</p> <p>This is sometimes a deterrent when shared with management who then decide reviews are too expensive or would take up too much time. This is an incorrect viewpoint.</p> <p>Investing effort in quality reviews has to be looked at in terms of the net benefits to be gained. As I mentioned earlier, effective use of quality reviews can certainly improve product quality, by delivering a product with fewer defects, but it can also result in significant cost and schedule reductions due to decreased rework.</p>

Critical Success Factors



- 1 • People must understand costs & benefits
- 2 • Must use an *effective* review process
- 3 • Review process must contain a defect removal step
- 4 • Reviewers must be trained

So...how come everyone's not doing reviews, given all the potential benefits they can bring to a project?

Well...I can tell you from my own experience that a major reason for this is a failure to understand how the costs and benefits interact...what I referred to earlier as the "physics" of quality reviews...the investment in review effort gets paid back several times over.

As I also said earlier...just doing quality reviews is not sufficient...the reviews must be effective. In a later lecture I'll describe what needs to be done to make them effective. In fact, unless reviews are effective they will add to the project cost and they will cause schedule increases.

Another critical success factor is that once detected, defects must be removed...meaning work products must actually be corrected. A significant portion of the review processes I have looked at for my clients did an okay job of detecting defects, but didn't incorporate a follow-up step to ensure that the detected defects had actually been resolved. While this may seem surprising, I have seen this situation far more frequently than I'd have like to...and it makes the quality review investment a waste of time and resources.

Finally, project team members need to have at least some minimal level of training to make this work at the project and certainly the organizational level. Knowing how to do effective reviews is not at all intuitive...if it was everybody would be doing them and doing them well. It doesn't take much in the way of training...a day is usually sufficient.

A few years ago I put together a one-day training session for a large insurance company client as a pilot project to demonstrate how effective quality reviews can produce significant benefits. The core teams that received the training and applied it to real projects saw immediate positive improvements. As a result, the company required that every IT professional take the training program. Over the course of several years I trained more than 500 of their staff.