stop lethal (defect) injection!

Agile Specification Quality Control

by Tom Gilb

INTRODUCTION

If we do specification inspections properly [2], the cost is barely tolerable for some: about one hour of effort, per page checked, per engineer. The harvest, if we are skilled, is between 40%-60% of major defects identified. The rest are not found yet, but they will be found in the final product, in testing, or released products. Finding many defects earlier than the test stage is beneficial and may even pay off. But there is a better way, which will appeal to many organizations that have not been able to stomach the high costs, and low effectiveness, of conventional inspection.

The main concept is to shift emphasis from finding and fixing defects early (in engineering specs before using them for construction) to estimating the specification defect density and using this information to motivate engineers to learn to avoid defect injection in the first place. This shift permits a dramatic cost savings. We can sample rather than check 100% of the specs when our purpose is measurement rather than "cleanup."

The main purpose of Agile Specification Quality Control (SQC) is to motivate individuals to reduce major specification defect insertion. Secondary SQC purposes are:

- To prevent uneconomic major-defect density specs from escaping downstream — and thus to avoid consequent delays and quality problems. The major tactic here is an SQC-determined numeric specification process exit barrier, such as "maximum 1.0 majors per page."
- To teach and reinforce current specification standards.

PROCESS DETAILS

The old inspection method was based on the idea of checking 100% of all pages, optimum rate checking (one page per hour), for teams of review engineers (two to five engineers). The maximum inspection process yield of major defects was and is in the range of 40%-80% depending on specification type (e.g., 60% for software source code, 80% for requirements [more likely 30%, since malpractice is common]). The reported ability to actually correct major defects was only five out of six attempted [1, 2]. All this amounts to the same order of magnitude of defects remaining as before the quality control process was applied. There is little or no change in the defect insertion density. In requirements specs, this regularly (by my field measures, for years) exceeds 100 major defects per 300 lines of specification.

The new "Agile method" is based on sampling the engineering specification:

- A few pages at a time
- Perhaps early (first 5% of a large volume)
- Continuously (every week or so) until the work is completed
- For each individual engineer (each one must be motivated and trained personally)

The sampled pages will be checked against a small set of about three to seven rules. For initial checks, these are usually as simple as "Clear enough to test, unambiguous to intended readers, no design options in the requirements." Checkers are asked to identify all deviations from these rules. These are "spec defects." The checkers classify any spec defect that can potentially lead to loss of time or product quality as "major" and report all major defects to a review leader.

The entire checking session might use only two engineers for 30-60 minutes. Once the session is over, the estimated number of defects actually present is calculated, based on the total found by the team. Generally speaking, the team

will be about one-third effective; so the estimated true number of majors per page is about three times the number of "unique majors" found by the team. This a rough engineering calculation, but it seems to work well in practice.

Exit Control

A prearranged standard is set for unacceptable major defect density. Initially the fail-to-exit level can be set at "more than 10.0 majors per page." In the longer term — say, beyond six months of culture change — you should be aiming to set the limit at more than 1.0 majors per page. For comparison's sake, IBM reported a maximum 0.25 defects per page [4]. At first the limit you set has to do with getting better as fast as humanly possible. Ultimately it is a matter of finding the level that pays off for the class of work you are doing.

Process Limitations

Please note that there are several limitations to this simplified process:

- It does not directly deal with process improvement.
- It uses only a small sample, so the accuracy is not as good as with a full or larger sample.
- The team may not have time or the experience to get up to speed on the rules and the concept of a major defect.
- A small team of two people does not have the known effectiveness of three or four people.
- You will not have the means of making corrections to the entire specification

Another consideration is that the checking will not have been carried out against all the possible source documents. Usually in the simplified SQC process, no source documents are used and memory is relied on. While this considerably speeds up the process, it does mean that the checking is not nearly as accurate. Nevertheless, if the sample turns up an estimated defect density 50 to 150 major defects per page (which is quite normal), that should be more than sufficient to convince the participants and their managers that they have a serious problem.

It may surprise you that the immediate solution to the problem of high defect density is neither to remove the defects from the document nor to change the corporate process. Instead, the most effective practical solution is to make sure each individual specification writer takes the defect density criteria (and their "no exit" consequence) seriously. They will then learn to follow the spec writing rules and, as a result, will reduce their personal defect injection rate. On average, a personal defect injection rate should fall by about 50% after each experience of the SQC process.

A MORE FORMAL PROCESS DESCRIPTION OF AGILE SQC

Simplified SQC Process

Tag: Simplified SQC. Version: 7 October 2004. Owner: Tom@Gilb.com. Status: Draft.

Entry Conditions

- A group of two or more suitable people[1] to carry out simplified SQC is assembled in a meeting.
- These people have sufficient time to complete a simplified SQC. Total estimated time: 30 to 60 minutes.
- There is a trained SQC team leader at the meeting to manage the process.

Procedure

P1: Identify Checkers. Two people, maybe more, should be identified to carry out the checking.

P2: Select Rules. The group identifies about three rules to use for checking the specification. My favorites are clarity ("clear enough to test"), unambiguous ("to the intended readership"), and completeness ("compared to sources"). For requirements, I also use "no design."

P3: Choose Sample(s). The group then selects sample(s) of about one page in length (300 noncommentary words). Choosing a page at random can add credibility, so long as it is representative of the content subject to quality control.

P4: Instruct Checkers. The SQC team leader briefly instructs the checkers about the rules, the checking rate, and how to document any issues and determine if they are majors.

[1] A suitable person is anyone who can correctly interpret the rules and the concept of a "major" spec defect.

P5: Check Sample. The checkers take between 10 and 30 minutes to check their sample against the selected rules. Each checker should mark up his copy of the document as he checks it, underlining issues and classifying them as "major" or not. At the end of checking, each checker should count the number of "possible majors" he has found on his page.

P6: Report Results. Each checker reports to the group the number of "possible majors" she found. The SQC team leader leads a discussion to determine how many of the possible majors are actually likely to be majors. Each checker determines her number of majors and reports it.

P7: Analyze Results. The SQC team leader extrapolates from the findings the number of majors in a single page (about six times[2] the most majors found by a single person or, alternatively, three times the unique majors found by a two- to four-person team). This gives the major defect density. If using more than one sample, he will average the densities found by the group on different pages. The SQC team leader then multiplies this average majors-per-page density by the number of pages to get the total number of major defects in the specification.

P8: Decide Action. If the number of majors per page found is large (10 majors or more), then the members of the group have to determine how they are going to get someone to write the specification properly. There is no economic point in looking at the other pages to find "all the defects" or correcting the majors already found. There are too many majors not found.

P9: Suggest Cause. Choose any major defect and think for a minute why it happened. Then jot down a short sentence, or better still a few words, to capture your verdict.

Exit Conditions

Exit if there are fewer than five majors per page extrapolated total density, or if an action plan to "rewrite" has been agreed.

AGILE SQC IN ACTION

At a Jet Engines Manufacturer

At one client, a maker of jet engines, we sampled two pages of an 82-page requirements document. Four managers checked page 81, and another four managers checked page 82. These two pages involved the "nonfunctional" requirements (security, etc.).

We agreed to check the pages against the following rules:

- Unambiguous to intended readership
- Clear enough to test
- No design specs mixed in

We agreed that violation of any one of these rules constituted a spec defect, and we set a spec exit level of "no more than 1.0 major defect per page." The managers were given 30 minutes to check their respective pages. At the end, the four managers who had checked page 81 had found 15, 15, 20, and 4 major defects, respectively. The members of the page 82 group found 24, 15 , 30, and 3 majors each.

From this data, we could determine the number of unique major defects found by the team. We could either log unique major defects (at three minutes each, it would be a three-hour job using nonagile methods) or estimate the result approximately. All managers chose the latter option.

To estimate the number of unique majors (nonduplicate, not counting as more than one, the same defects found by two or more checkers), we can estimate by doubling the count of the largest amount found in a small group. This approach is based on observations done at Cray Research [2], and it works well. In this case, it means that the page 82 group found about 60 (i.e., 2 x 30) unique majors per page. The page 81 group had about 40 total unique major defects they could log if they so chose.

But as I observed earlier, checkers do not find 100% of the majors defects present — they find about one-third. We can easily prove that this is true. Begin by removing the major defects you have identified. That should leave twice that

[2]The reason we multiply by 6 is that we have learned through experience [2] that the total unique defects found by a team is approximately twice the number found by the team member who uncovers the most defects. We also find that inexperienced teams using simplified SQC seem to catch about one-third of the major defects that are actually there. So 2 x 3 = 6 is the factor we use (or 3x the number of unique majors found by the team).

number remaining. This sounds incredible. How could people miss so many on a single page? The proof comes when you repeat the checking process and predictably find one-third of the remainder — and can prove they were there on the first checking pass. Skeptics turn into believers at this point.

In this case, the managers accepted my assertion that the 60 majors they found on page 82 were an indication of about 180 majors on the page (and 120 majors on page 81, positing the same density as page 82). This means that the document had an average of 150 — that is, (120 + 180)/2 = 150 — majors per page. I asked the managers if they felt this was probably typical for the other pages covering the functional requirements. They said (and all managers do say) they had no doubt that it did. This meant that if the requirements document had an average of 150 majors per page on 82 total pages, it contained 12,300 major defects overall!

Now, it's important to note that not all major defects in specs lead directly to bugs. Two pieces of research I recall showed that 25% to 35% of the majors actually turn into bugs. I have found that a good rule of thumb is that one-third of the major defects will cause bugs in the system. This implies that 4,100 (i.e., 12,300/3) bugs will occur in the system in question.

One of my clients (Philips Defence, UK; see [2]) studied about 1,000 major defects found in spec inspection (from a wide variety of engineering specs, not just software) and discovered that the median downstream cost of not finding them would have been 9.3 hours each. So I use 10 hours as a rough approximation of the cost of fixing a bug downstream in the test and field stages.

For my jet engine manufacturing client, that suggests 41,000 hours of effort lost in the project through faulty requirements. I was quite shocked at the implication of this quick estimate based on a small sample, but the managers were quite at home with it. "Don't worry, Tom," they said, "we believe you!"

"Why?" I said.

They replied, "Because (and we know you did not have any inkling of this) we have 10 people on the project, using about 2,000 hours/year, and the project is already one year late (20,000 hours). And we have at least one more year of correcting the problems before we can finish!"

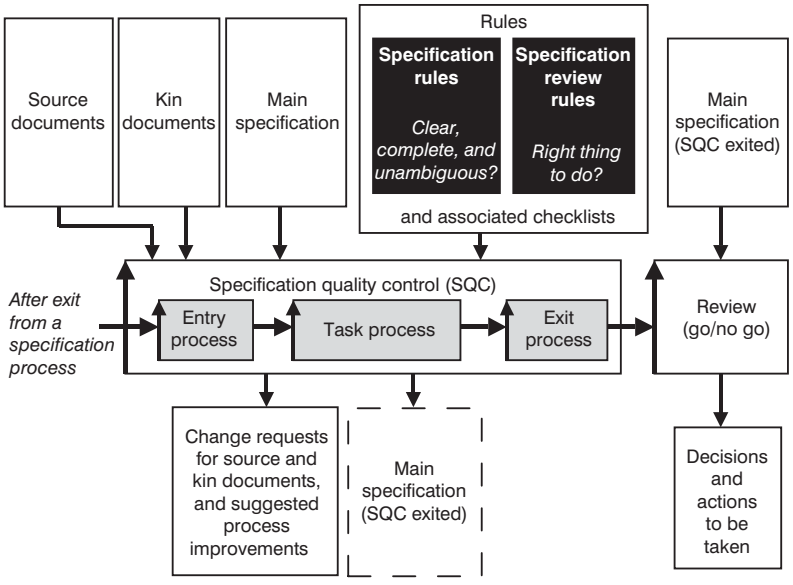

Figure 1 — The SQC process: a model that applies both with and without "sampling"— the Agile option [3].

In an Air Traffic Control Simulator

A client had a seriously delayed software component for an air traffic control (ATC) simulator. The contract dictated about 80,000 pages of logic specifications. The supplier had written and approved about 40,000 pages of these. The next stage for the logic specs was writing the software.

The divisional director gave me the technical managers for a day to try to sort out the problem. These men had all personally signed off on the 40,000 pages of specs. We pulled three random pages from the 40,000, and I asked the managers to find logic errors in the specs — errors in the sense that, if coded, the ATC system would be wrong. Within an hour of checking, they as a group had found 19 "major

defects" in the three sample pages, pages they agreed were representative of the others.

That evening, the director took 30 minutes to check the 19 defects personally, while his managers and I waited in his office. He finally said, "If I let one of these defects get out to our customer, the CEO would fire me!

Now, the 19 defects found in the three pages really represent about three times that. The managers probably did not find two-thirds of the existing defects. So the managers had signed off on about 0.76 million bugs (19 x 40,000), and they had only done half the contracted logic specification. Clearly, the sample told us a lot.

We got to thinking that afternoon about what could have been done better. The conclusion was that we had a "factory" of analysts producing about 20 major defects per page of ATC logic specification. We also concluded that if we had taken such a sample earlier — say, after the first dozens of pages were written — we might have discovered the defect density and done something effective about it.

I asked one manager, whose signature was the third one on the spec approval, why he signed off on what we all acknowledged was a tragedy. He told me it was because "the other managers signed it ahead of him." I guess that is when I lost faith in management approvals.

It's too bad that this company did not have Agile SPC as a practice. The project got completed, but only after being sold off to another industry. The director lost his job, and it was not just for a single defect. His corporation, I later realized, had a bad ingrained habit. They did not review specifications until they were all completed.

The approach we finally successfully used to move the project out to the customer was evolutionary delivery — even though my client initially said that it could not be done "because it was not in the contract."

CONCLUSION

Agile SQC costs very little, but it can drive defect injection down by an order (and then, with time, two orders) of magnitude. The key SQC concept compared to traditional software inspection methods is to measure by sampling and use the information to motivate people to "learn the rules" (standards, best practices) and reduce their defect injection rate.

Conventional inspection techniques are doomed to high costs and low impact because they can only hope to find about half the problems, and they can only do that by spending around three to four hours of engineering effort per page on all pages of specification. For many organizations, Agile SQC may prove a better alternative.

REFERENCES

1. Fagan. M.E. "Advances in Software Inspections." *IEEE Transactions on Software Engineering*, Vol. SE-12, No. 7, July 1986, pp. 744-751.

2. Gilb, T., and D. Graham. *Software Inspection*. Addison-Wesley, 1993.

3. Gilb, T. *Competitive Engineering*. Butterworth-Heinemann, forthcoming 2005.

4. Humphrey, W.S. *Managing the Software Process*. Addison-Wesley, 1989.

Tom Gilb is a consultant, author, and teacher. He has published nine books and numerous papers. Immediately forthcoming is his latest book, Competitive Engineering.

Mr. Gilb primarily works at changing systems engineering cultures in large multinational corporations. His major technical interests are in the areas of requirements engineering, design and architecture, evolutionary project management, and specification quality control (inspection). His clients include McDonnell Douglas/Boeing, BAE Systems, HP, Nokia, Sony/Ericsson, Philips, CitiGroup, Intel, Microsoft, Canon, and United Defense. He does pro bono work for US DoD, UK MoD, various charitable organizations, and in developing countries, such as India, China, and Korea.

Mr. Gilb started working for IBM in 1958. He stayed at IBM for five years and has been an independent consultant since then. He was born in California and lives in Norway.

Mr. Gilb can be reached at Tom@Gilb.com; Web site: www.gilb.com.