Design reviews would benefit from the support of formal rules. By the use of relevant rules, it should be possible to ensure, prior to a review, that all relevant information for the review is present in the design specifications, and that all the minimum review criteria are met. This will ensure management time is not wasted and aid better decision making. This article recommends that the specification quality control (SQC) method be used to do this additional quality control. In addition, this article outlines the impact of evolutionary project management (Evo) on the design review process.

Key words: evolutionary project management, peer review, quality control of specifications, review, review standards

SQP References

Identifying Code-Inspection Improvements Using Statistical Black Belt Techniques
vol. 6, issue 1
Jeff Holmes

Using Capture-Recapture Models for the Reinspection Decision
vol. 5, issue 2
Julie Barnard et al.

Case Study of an Improvement Program Featuring Reviews and Inspections
vol. 3, issue 3
Robert MacFarland

Planning to Get the Most Out of Inspection
vol. 2, issue 2
Tom Gilb

Document Inspection as an Agent of Change
vol. 2, issue 1
Yingxu Wang and Alec Dorling et al.

TALKING POINTS

Rule-Based Design Reviews

TOM GILB

INTRODUCTION

It is known that most projects are a partial or total failure, and few are totally successful (Morris 1994; Taylor 2001; Johnson 2001). A key reason for this lack of success is the failure of design reviews. In other words, designs must have been approved that were in some way inappropriate.

Based on reading the literature and participating in numerous requirement and design inspections in many different industries internationally, I fear that most design reviews are carried out:

- On poorly written design specifications (in the sense that the design specifications lack sufficient detail for them to be safely evaluated)
- Using highly polluted requirement specifications (requirements being the basis for evaluating any design).

I find it unacceptable that design reviewers are given *no quantified knowledge* of the quality of the design specification, or of the estimated ability of the design(s) to impact on the requirements (that is, the system performance and costs). A common underlying problem is that specification of design is carried out on the basis of inadequate design standards. I suggest the following remedies:

- A high, defined standard of requirements should be met before entry to the design process itself is permitted.
- Design specification should initially pass quality control against design specification rules to ensure the designs are clearly and fully described.
- Designs should be specified in enough detail to accurately estimate their dominant performance and cost characteristics.
- A set of designs should be seen to credibly and numerically contribute to meeting the requirements (the set of performance targets within the resource budgets).
- The design review process should work in the context of evolutionary cycles of design (for example, in 50 steps), and not operate on a large monolithic total initial design set.

In other words, this article recommends that design specifications undergo specification quality control (SQC) to establish their conformance to the *standards for design specification*, and on successful exit from that, undergo a second SQC to determine their conformance to the *standards for design review* (see Figure 1). Only on successful exit from this second SQC should the actual review, by a body of senior people, take place. One should not waste their time on substandard design proposals.

This recommendation for a second SQC represents an extension of the current practices. It suggests that the use of SQC be expanded to specifically assist the various classes of review. The idea is that SQC can assist reviews by checking that all the information required by the decision makers, that is available, has been gathered and processed as necessary, and that the minimum criteria required to take design specifications forward to review has been met.

Note that this article does not explain how the impacts of designs on the requirements are estimated or how tradeoffs are made. (Impact estimation is a process for achieving this; see Gilb 1998.)

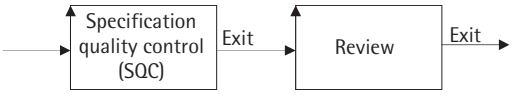
TERMINOLOGY

Specification quality control (SQC): SQC is also known by the name "inspection." Given that inspection has another meaning to engineers within manufacturing, I prefer to use the term SQC and avoid the collision of using a term with two different meanings. For example, Boeing refused to use the term inspection and used process error prevention (PEP) instead. SQC is also sometimes termed "peer review." I do not use this term because I want to make a clear distinction between "quality control" and "review."

SQC is a rigorous quality control discipline concerned with defect detection, defect measurement, defect removal, process improvement, and entry/exit controls. It is based on evaluating specification conformance to specification rules.

Traditionally, SQC does not pretend to judge the specifications in terms of their relevance or profitability in the real world. It is primarily concerned with making sure that the specifications are clear, complete, and consistent by checking a specification and any of its source and kin documents against "specification rules." It judges whether the specification is suitable to be used in subsequent engineering or management processes. However, by using a different type of rules, "specification review rules," it is possible to extend the SQC process to checking the readiness of specifications for review. This could be for a business review, a progress review, or a technical review. See Figure 2 for an outline of the extended SQC process.

FIGURE 1 The two necessary distinct processes



- Specification quality control (SQC) – Is it following the standards (rules)?
- Review – Is it the right stuff?

Review: A review is any process of human examination of ideas with a defined purpose and defined standards of inquiry.

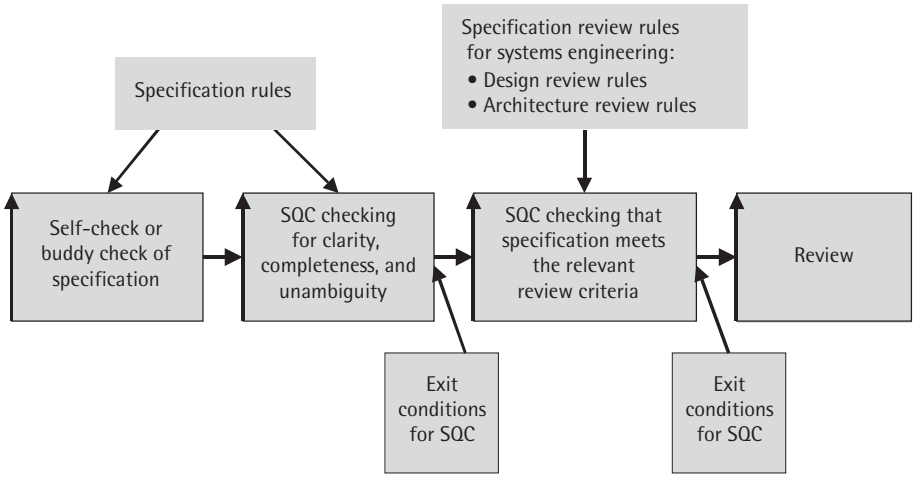
Design specification: A design specification is the written specification of a design idea. A set of design specifications attempts to solve a design problem. Identification and documentation of the individual design ideas, and their potential contribution toward meeting the requirements, helps selection of the "best" design ideas for implementation.

The design specifications should contain information about the *expected* attributes of the designs for meeting requirements. This 'expected attributes information' of a design specification might be expressed in the form of an impact estimation table, or it can be as simple as an assertion of impacts on requirements, referenced by their tags (see example in Figure 3).

This article is presented in terms of a set of design review principles.

- DR1: If the specifications are unclear, one is not ready to review whether the design is the "right thing" to do.
- DR2: Individuals cannot review designs if they are unclear whether each specific design is mandatory (a design constraint, which is a requirement) or optional (one of many possible design solutions).
- DR3: Design impacts must be calculated and submitted to a review; they cannot systematically be developed *during* the review.

FIGURE 2 SQC and review processes. Review should follow successful exit from SQC



- DR4: It is part of the purpose of a design review to identify any defective information in the design impact estimates.
- DR5: Certain objective review criteria must be met prior to actually carrying out a review; otherwise, the review may be wasted.
- DR6: The review process should not wait until a large amount of specification is completed; sampling reviews should be held early, and frequently, to spot systemic problems early.
- DR7: A design review process should be carried out on a realistic amount of information and conducted at an effective rate. Don't so overwhelm reviewers that they become incapable of spotting problems.
- DR8: To benefit from feedback, the design review process should be done evolutionarily, as a long series of design reviews, each review deciding on the next planned evolutionary step.
- DR9: The real purpose of design reviews is not to approve a design as correct, but to uncover specific risks and specification defects.

These principles are not scientific laws. They are observations and opinions based on decades of personal experience and empirical data collected and published by clients. They are simply suggestions as to useful ways to think about reviews.

SQC PRINCIPLES

Principle DR1: If the specifications are unclear, one is not ready to review whether the design is the "right thing" to do.

Have you ever actually counted the quantity of unclear and ambiguous expressions per page in a real specification? I get my clients to do this using SQC almost every week. The result is consistent and always provides an element of shock.

I define a specification defect as any violation of a set of specification rules. As a simple introduction to SQC, I ask a team of two to five people to select a random page of their "typical" specification and then spend about 15 minutes checking it individually, applying these two rules:

- Rule 1: The specification must be unambiguous to all in the intended readership.
- Rule 2: The specification must be clear enough to test for correct implementation.

For the first rule, it is not a matter of whether the people looking for defects understand the specification; they must role-play the weakest link in the set of people who might have to correctly understand it (for example, "Imagine you had just started here right out of school," or that "You were an Indian subcontractor in Bangalore"). They need to check that there is only one possible plausible interpretation.

FIGURE 3 Example of a design specification showing some impacts (though no specific numeric estimates) for the design on the requirements

Engineer motivation:
Gist: Motivate and use of free time off
Type: Design idea
Impacts (Objectives): (engineering productivity, engineering costs)
Impacts (Costs): (staff costs, available engineering hours)
Definition: Offer all engineers up to 20 percent of their normal working hours per year as discretionary time off to invest in health, family, and knowledge (studies, write papers, go to conferences)
Source: Productivity Committee Report 1.4.3
Implementor: Human Resources Director

I also ask the participants to judge whether the rule violations they find (the "specification defects") are major (or minor). Defects are major if it is judged *possible* (as opposed to *im*possible) that the faulty specification *could* cause something wrong to happen during systems engineering or in the operational system (for example, product faults, test case faults, procurement errors, estimation errors, software bugs, and delays).

Note that if the rules are strongly formulated, then it is less critical whom one selects to be a checker. Ideally, one can use less experienced people to carry out the SQC checking, and then bring in the more experienced people to conduct the actual review. This is because the experience of what to look for is built into the rules and corresponding checklists (of how to interpret the rules). The actual number of checkers is not critical from the point of view of measurement of conformance to standards. It does make a measurable difference, but this can be considered in adjusting the estimates of major defects per page.

Most people (for example, the writer of the specification and his or her peers) consistently manage to find between five and 15 defects in a single sample page of a specification within the 15 minutes.

From experience, I know that for a small team of two to five checkers, doubling the number of majors found by the checker who finds the most majors tends to reveal the total number found by the entire team. I also know that the initial effectiveness in such a situation for finding majors is in the range of 25 percent to 35 percent. (This is partly due to the time allowed, and partly due to the source information available.) For the sake of simplicity, say it is 33 percent.

So if 15 majors were found, I would estimate that there were 15 x 2 x 3 = 90 defects on the single sample page. This means there are about 90 potential misunderstandings of a specification per page. This is "normal" technical specification "pollution," but it is not a good or necessary basis on which to decide if the design itself is a good one. Any one of those major defects alone is capable of corrupting one's understanding of a whole design, and capable of making one's judgments, in a design review, worthless.

The bad news is that this level of majors per page will continue to persist unless management changes the game. The good news is that it is possible to bring down the average level of majors per page by two orders of magnitude within weeks of doing the right things.

One can bring down the majors per page density by stating the specification rules (such as "clear" and "unambiguous"), and then training individual engineers to follow these simple rules. One does that by teaching the rules, and then making sure that rule violations are measured. Work is considered unacceptable when a defined level of pollution (such as one major per page) is exceeded. Under such conditions, an individual will learn at a rate of about 50 percent defect injection reduction per cycle of specify/check/correct (see Figure 4). We proved this initially in 1988 at Douglas Aircraft for hundreds of engineers working on airplane drawings. We repeated it the year after at Boeing, and later we experienced this same improvement rate at Ericsson. It doesn't matter where or with which technology this is done, it works. Others, such as Raytheon Defense Electronics, have reported the same experience working within military software projects (Haley et al. 1995).

What does all this mean for design reviews? In summary, all specifications need to be quality controlled to make sure they are not polluted before one can seriously undertake a design review.

I suggest the need for at least three entry conditions into a design review:

- All requirement specifications have successfully exited on a numeric basis from SQC.

- All design specifications have successfully exited on a numeric basis from SQC using design specification rules.
- All design specifications have successfully exited on a numeric basis from SQC using design specification review rules.

I recommend setting the numeric level at "less than 1.0 estimated remaining major defect/page." Anything less is wasting the review team's time and project time. What quality levels does the current review process demand in its entry process?

Principle DR2: Individuals cannot review designs if they are unclear whether each specific design is mandatory (a design constraint, which is a requirement) or optional (one of many possible design solutions).

What most people title "Requirements" often includes some design specification, which is not actually part of the requirements. Only *design constraints* should be specified as part of the requirement specification: any design that is optional has no place there; it should be only in the design specification. (Requirements are the end states needed by stakeholders, irrespective of implementation detail. Design is a decision about the means of implementing those requirements in practice.)

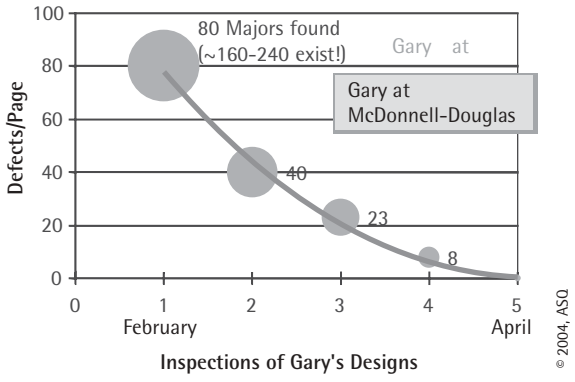
Of course, mandatory designs (the required design constraints) must also have detailed design specifications. Individuals need to ensure they have the design specification for any mandatory design (clearly marked as mandatory), and that it is considered alongside the optional design.

Principle DR3: Design impacts must be calculated and submitted to a review; they cannot systematically be developed *during* the review.

There is no time in a design review to begin a process of collecting facts and making estimates about the multiple impacts on performance and costs of each design. The proper time to do that is before the review, not during it.

If the estimates are not made, this fact should be caught in the SQC preceding the review. Note that such SQC merely observes that such estimates are not made, or not made properly (for example, they lack any evidence).

FIGURE 4 The individual engineer is generally capable of reducing the defect injection density by 50 percent per cycle of personal learning and feedback



Principle DR4: It is part of the purpose of a design review to identify any defective information in the design impact estimates.

See Figure 5 for an example of an impact estimation table containing some design impact estimates. (See Gilb 1998 for further details on the impact estimation process.)

The design reviewers are allowed to question the accuracy of the impact estimates, and the evidence and credibility claimed. The design reviewers are experts in their fields and should consider if they agree with the presented data.

SQC carried out prior to a design review is mainly concerned with finding out that the required impact estimation process for designs was apparently performed.

Principle DR5: Certain objective review criteria must be met prior to actually carrying out a review; otherwise, the review may be wasted.

In addition to making sure the design engineer has adhered to the design specification rules (see Figure 6), there needs to be a check against design specification review rules in preparation for a design review. Only if the design meets the design review rules should a review proceed.

FIGURE 5 This is a conceptual example. Three goals (performance targets) and two resource targets are having real (% of impact needed to reach the target) impacts on the performance and cost attributes, and are tracked as steps are delivered. The same table is also being used to specify the impact estimates for the future planned steps. So at each step, the project can learn from the reality of the design impacts included on a step, and the deviation from design impact estimates.

Step Target Requirement	**Step 1** Plan % (of target)	Actual %	Deviation %	**Step 2** to **Step 20** Plan %	Plan % cumulated to here	**Step 21** [CA, NV, WA] plan %	Plan % cumulated to here	**Step 22** [all others] plan %	Plan % cumulated to here
Performance 1	5	3	-2	40	43	40	83	-20	63
Performance 2	10	12	+2	50	62	30	92	60	152
Performance 3	20	13	-7	20	33	20	53	30	83
Cost A	1	3	+2	25	28	10	38	20	58
Cost B	4	6	+2	38	44	0	44	5	49

Principle DR6: The review process should not wait until a large amount of specification is completed; sampling reviews should be held early and frequently to spot systemic problems early.

Once on a German Air Traffic Control project carried out in Sweden, I saw the signatures of seven managers approving the detailed logic of the air traffic management: 40,000 pages of logic design (the next stage was the programming). Later that day, using the sampling process described earlier, they found 19 major defects in a random representative sample of three of the 40,000 pages of logic design. This was, of course, about one-third of what was actually present in the pages. Their divisional director took 30 minutes to personally check the 19 majors while the eight of us waited in his office, and agreed that they were serious. At about 20 majors per page present, that meant there were about (20 x 40,000) 800,000 majors *approved* for coding by the management review committee. I asked signature no. 3 on the list why he signed off on what we now recognized was a very polluted document. His reply will not surprise the experienced reader: "Because the other two signed it before me." This is why I am skeptical about review committee approvals!

We had many interesting discussions on the basis of this finding. Mainly, if they had bothered to do some simple samples early, for example, after 100 pages had been written, they might have been able to prevent most of this work from being totally wasted. In other words, if reviews had been carried out earlier, and if they had demanded numeric quality controls were in place, then it is unlikely that the defect injection would have been allowed to continue at such a level.

So there is a lesson to be learned here about early sampling of partial work. Don't wait for the entire specification before doing some basic quality control on it—a point I will return to toward the end of this article.

DR7: A design review process should be carried out on a realistic amount of information and conducted at an effective rate. Don't so overwhelm reviewers that they become incapable of spotting problems.

If an attempt is made to review too much design at once, then the analysis is unlikely to be done sufficiently, and the truth is likely to be obscured. If 40 or more (try 40,000 pages, as in the previous example) of design specifications are delivered at once to a design review, then the review group will not have time to study, discuss, or criticize anything in detail. The reality of risks and problems will be lost and, as a result, the review team will not learn about its misjudgments. The project or product will probably fail and the team will not be clear about the cause of failure.

I suggest only feeding a small volume of ideas into a design review committee so that:

- The committee has a chance to do its work properly.

- One can better ensure that any preparatory work has been done properly.

I would suggest one page of design per committee hour as a rough guide.

Deciding what is a realistic sample is the responsibility of the SQC team leader. Of course, he or she should ask questions of the author and the design team to ensure they have a representative sample. One key point to grasp is that if a small sample shows a high defect density, then it is not worth continuing – there clearly is a problem that the document author has to address. Of course, if the sample is "clean" that doesn't imply the entire specification is fine. It does demonstrate, however, that the author knows and conforms to the design specification rules.

Principle DR8: To benefit from feedback, the design review process should be done evolutionarily, as a long series of design reviews, each review deciding on the next planned evolutionary step.

Ideally, most projects should be carried out using evolutionary project management (Evo). The critical distinction between Evo methods and their generic cousins iterative and incremental is that evolutionary processes (which are both iterative and incremental, too) gather facts about impacts, analyze those facts, and change behavior, if necessary, in order to succeed in the higher-level objectives.

Evo means testing the stakeholder effects of a design idea in the field. I suggest maybe 50 ideas, one a week for a year of development effort. The design review committee becomes a learning process; that is, the review team will benefit from the Evo cycle experience feedback concerning the implementation of each previous design, and the team will learn quickly and realistically, by real experience, how to evaluate designs.

DR9: The real purpose of design reviews is not to approve a design as correct, but to uncover specific risks and specification defects.

Design reviews are about risk analysis, and one must build a much better foundation in order to reliably carry out that function. This is where Evo methods are helpful. If individuals do not implement their designs evolutionarily, they might never learn that a particular one of them was their downfall. Even if they do learn which one it was, it is probably too late to do anything about it on that particular project.

FIGURE 6 Some examples of design specification rules. These would be used in an SQC of a design specification. Rule AR6 would likely be expanded into several distinct rules. Only if the defect level was sufficiently low (say, less than one remaining major defect per page) would the specification be submitted for further SQC to see if the specification was ready for review.

Examples of Specification Rules for Design

AR1: Cost detail: The architecture must be specified in enough detail to permit at least correct order of magnitude impact estimation for costing.

AR2: Cost estimates: Estimates must be made and included as to the order of magnitude of all critical costs of the architecture (particularly for those resources that are budgeted requirements).

AR3: Performance detail: The architecture specification must include enough detail to allow order of magnitude correct estimation of the specification's impact on all specified performance goal levels (that is, all the qualities, resource savings, and work capacity requirements).

AR4: Performance estimates: Estimates will be included for the impacts on all the critical performance requirements, at correct order of magnitude.

AR5: Background detail for estimate: Each impact estimate must be supported by:
- The factual experiential evidence for the estimate
- The source of these facts
- The uncertainty boundaries/error margins (±%)
- A credibility rating (on a 0.0 to 1.0 scale)

These data ideally will be presented using an impact estimation table.

AR6: Additional data: The architectural specification must include additional specification as detailed in the current architecture specification template. This will include stakeholders, detailed design reference (if any), QC level, review approval level, risks, issues, and dependencies.

If individuals are using Evo, a review could even deliberately decide to implement a high-risk step to find out the results. It is the benefits obtained that count; total avoidance of risk is not the aim!

The key benefit of more frequent reviews is that risk is made more manageable. This might mean that lower-level management can be empowered to make

the review decisions. Alternatively, it could mean the design review itself might become obsolete, since reality is going to give designers more reliable advice than any committee of experts.

IMPLEMENTING SQC

The costs of SQC So what are the costs of implementing SQC? In an organization already using SQC, the additional costs would be in deriving the specification review rules and carrying out the additional SQCs.

An organization unfamiliar with SQC could refer to Chapter 2 of the book *Software Inspection* (Gilb and Graham 1993). The main message is that the measurable benefits of using SQC outweigh the direct costs; for example, Haley et al. (1995) report benefits outweighing costs by 8 to 1. The direct costs are the training of the SQC team leaders, which takes about a week, and the operational costs of running the SQCs, which is about one hour of review per page (300 words) of text per reviewer.

For effective reviews, four to five reviewers should carry out the checking. For efficient reviews, two or three reviewers should be used. SQC should always be cost effective; otherwise, it should not be used.

Checking rate One critical factor for successful SQC is that the correct checking rate must be adhered to. If SQC is carried out at too fast a checking rate then some defects that should be found will not be found, which will give a spurious quality level. Too slow is equally bad, as that will make the SQC less cost effective.

Rate of quality improvement SQC can reduce defect density by one to two orders of magnitude (from 100-plus to 10 to less than one per page) in six to 30 months for an organization, but it operates at the level of motivated individuals. New employees have to go through the process from their initial level of technical writing capability in relation to the defined rules that apply.

Level of quality improvement A typical and simple example of the reduction of defect densities, which was cited in (Haley et al. 1995), was a factor of about three. Many other studies confirm such substantial reductions, for example, Hewlett-Packard (Grady and Van Slack 1994) and Ericsson (MacFarland 1998).

Management support One key factor required for successful SQC is strong support for SQC from management. Quantified exit criteria must be set and adhered to, such as a maximum of one remaining defect per page.

Limitations of the defined rules The main limitation of the method is that it is limited by the defined rules. The ability to detect issues is dependent on the quality of the rules. One way of ensuring the coverage of the rules is to separate the different types of rules into different rule sets. That way it is clearer to everyone the extent of the checking in each different dimension; for example, one rule set can focus on design costs, another on the relevance of the design specifications to the requirements, and another on architectural considerations.

Deadline pressure People often are worried about introducing additional tasks into their systems development process, but SQC can actually help reduce schedule pressures by reducing rework! Using SQC at early stages in the development cycle means that some defects are removed earlier. This is beneficial because the later in the development cycle that a defect is detected, the more it costs to correct. Even better, by identifying the causes of repetitive defect insertion, one can put in place corrective actions (by process improvement), and actually eliminate certain defects. The documented reduction in rework is about 40 percent (Gilb and Graham 1993; Haley et al. 1995).

Agile or Simplified SQC

By using such techniques as sampling, SQC can be implemented in an agile systems development environment. I have developed a cut-down, simplified version of SQC, which has been successfully used at one of the largest financial institutions in the world. The initial results after six months were a reduction from on average 88 major defects per page to about 11 defects per page. The expectation is to reduce by another order of magnitude within the next one to two years.

The main ideas behind "agile" SQC are:

- A shift of focus from cleaning up all the defects on a page to defect density measurement by sampling
- A shift from defect correction to motivating specification writers to follow the defined rules
- Use of the measurement and exit process to motivate specification writers to change behavior

This cut-down version also has benefits for smaller organizations. The agile SQC version of the method is designed to work with two people and one single sample page, with an elapse time of approximately 30 minutes.

- The agile SQC version is presented in Figure 7. Note, there are several limitations to this simplified process:
  - It does not directly deal with process improvement.

FIGURE 7 The "agile" or simplified SQC process.

Agile or Simplified SQC Process

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

Entry conditions
- A group of two or more suitable people* to carry out Simplified SQC is assembled in a meeting.
- These people have sufficient time to complete a Simplified SQC. Total elapse 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. (Some favorites are clarity ("clear enough to test"), unambiguousness ("to the intended readership"), and completeness ("compared to sources").

P3: Choose sample(s): The group then selects sample(s) of about one page in length (300 non-commentary words). Choosing a page at random can add credibility, as long as it is representative of the content subject to quality control. The group should decide whether all the checkers should use the same sample or whether different samples are more appropriate.

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 major defects (majors).

P5: Check sample: The checkers use between 10 and 30 minutes to check their sample against the selected rules. Checkers should "mark up" their copy of the document as they check (underlining issues, and classifying them as major or not). At the end of checking, checkers should count the number of possible majors they have found in their page.

P6: Report results: The checkers each report to the group their number of possible majors. The SQC team leader leads a discussion to determine how many of the possible majors are likely to be majors. Checkers determine their number of majors and report it.

P7: Analyze results: The SQC team leader extrapolates from the findings the number of majors in a single page (about six times** 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, average the densities found by the group in different pages. The SQC team leader then multiplies this average major defects per page density by the total 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 there is little point in the group doing anything, except determining 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 give a short sentence, or better still a few words, to capture the verdict.

Exit Conditions
- Exit if less than five majors per page extrapolated total density, or if an action plan to "rewrite" has been agreed.

Notes:

* A suitable person is anyone who can correctly interpret the rules and the concept of "major."

** Concerning the factor of multiplying by 6 : We have found by experience (Gilb and Graham 1993) that the total unique defects found by a team is approximately twice that of the number found by the person who finds the most defects in the team. We also find that inexperienced teams using Simplified SQC seem to have about one-third effectiveness in identifying 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).

- It is only a small sample so the accuracy is not as good as a full or larger sample.
- The team will not have time or experience to get up to speed on the rules and the concept of major defect.
- A small team of two people does not have the known effectiveness of three or four people.
- One will not have the basis for making corrections to the entire specification.
- 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 means that the checking is not nearly as accurate, it does considerably speed up the process.)

However, if the sample turns up a defects density estimation of 50 to 150 major defects per page (which is normal), this is more than sufficient to convince the people participating and their managers that they have a serious problem.

The immediate long-term solution to the problem of high defect density is neither to remove the defects from the document, nor to change the corporate process. The most effective practical solution is to make sure each individual specification writer takes the defect density criteria (and its "no exit" consequence) seriously. They will then learn to follow the rules and, as a result, will reduce their personal defect injection rate. On average, a personal defect injection rate should fall by about 50 percent after each experience of using the SQC process. Widespread use of SQC will result in large numbers of engineers learning to follow the rules.

CONCLUSIONS

Any design specifications input into a design review must be of known high clarity and completeness: they should have successfully exited from SQC using both design specification rules, and then later using design specification review rules.

Design reviews should be held throughout the life of a system. They should be held at early stages on samples of the design work to ensure initial problems and misunderstandings are detected as soon as possible. Reviews should also be held at appropriately frequent intervals to avoid giving reviewers too much to review at one time.

For an evolutionary project, reviews should be held to decide/agree on each "next" evolutionary step. By utilizing feedback, reviewers will learn more about their mistakes and successes, and any actions required to correct/improve project progress can be taken.

A design review should not be an informal management meeting to collect unfocused opinions under pressure.

REFERENCES

Gilb, T., and D. Graham. 1993. *Software inspection*. Reading, Mass.: Addison-Wesley.

Gilb, Tom. 1998. Impact estimation tables: Understanding complex technology quantitatively. *Crosstalk*. (December). See http://www.stsc.hill.af.mil/crosstalk/1998/12/index.html/ .

Gilb, T. 2004. *Competitive engineering: A handbook for systems engineering, requirements engineering, and software engineering using Planguage*, draft manuscript on www.Gilb.com . To be published by Elsevier, January 2005.

Grady, R. B., and T. Van Slack. 1994. Key lessons in achieving widespread inspection use. *IEEE Software* (July): 46-58. This paper also appears in (Wheeler, Brykczynski, and Meeson 1996). This paper is available for a fee at http://csdl.computer.org/comp/mags/so/1994/04/s4046abs.htm .

Haley, T., B. Ireland, ed., Wojtaszek, D. Nash, and R. Dion. 1995. *Raytheon Electronic Systems experience in Software Process Improvement*, Raytheon. This paper is available online at http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.017.html/ .

Johnson, J., K. D. Boucher, K. Connors, and J. Robinson. 2001. Collaborating on project success. *Software Magazine* (February).

MacFarland, R. 1998. Improvement program spreading effective use of reviews and inspections (R&I) throughout the switching organization of a multinational telecommunications company. Presented at Eurostar 98 Software Testing Analysis and Review Conference.

Morris, P. W. G. 1994. *The management of projects*. London: Thomas Telford.

Taylor, A. 2001. IT projects sink or swim. *BCS Review*. See URL: http://www.bcs.org.uk/review/2001/articles/itservices/projects.htm/ .

BIOGRAPHY

Tom Gilb was born in Pasadena, Calif. He emigrated to London in 1956 and to Norway in 1958. He took his first job with IBM in 1958, and became a freelance consultant in 1960. He coined the term "software metrics," and is credited as the main inspiration for SEI CMM Level 4.

His interests include corporate strategy, requirements engineering, inspection, and evolutionary methods. He has worked with numerous multinational clients, including BAE Systems, Sun Microsystems, CitiGroup, Philips Medical Systems, Nokia, Ericsson, HP, Intel, Microsoft, British Airports Authority, Symbian, and Canon.

He is the author of nine books. More recent books include *Principles of Software Engineering Management* (1988) and *Software Inspection* (coauthored with Dorothy Graham in 1993). His next book, *Competitive Engineering* will be published in January 2005. Many of his current papers are available on his Web site, www.Gilb.com. Gilb can be reached at: Tom@Gilb.com .