Rule-Based Design Reviews:

Objective Design Reviews and Using Evolutionary Quantified Continuous Feedback to Judge Designs

Iver Holtersvei 2, NO-1410 Kolbotn, Norway, Tom@Gilb.com, www.Gilb.com, +47 66801697

Abstract. 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 the relevant information for a 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 paper recommends that the Specification Quality Control (SQC) method be used to do this additional quality control.

In addition, this paper outlines the impact of Evolutionary Project Management (Evo) on the design review process.

Introduction

We know that most projects are a partial or total failure, and few are totally successful (Morris 1994, Taylor 2001, Johnson 2001: Extracts from Extreme Chaos 2001, a Standish Report). I suggest that a key reason for this lack of project success is the failure of design reviews. In other words, that whatever designs did get approved, should probably not have been.

Based on reading the literature, and participating in a very large number of 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), and
- using highly polluted requirement specifications (Requirements being the basis for evaluating any design).

I find it unacceptable that a design review is 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). The underlying problem is that specification of design is carried out on the basis of inadequate design standards.

It is the point of this paper to suggest the following remedies:

- a high defined standard of requirements is met before entry to the design process itself is permitted
- design specification should initially pass quality control (SQC) against design specification rules to make sure the designs are clearly and fully described
- designs should be specified in sufficient detail to enable reasonably accurate estimation

of 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 design set

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. 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 or a technical review.

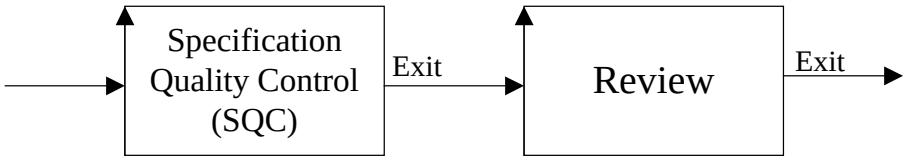

Figure 1: The two necessary distinct processes:

- **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 towards 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 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 2).

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% 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.

Figure 2: Example of a design specification showing some impacts (though no specific numeric estimates!) for the design on the requirements.

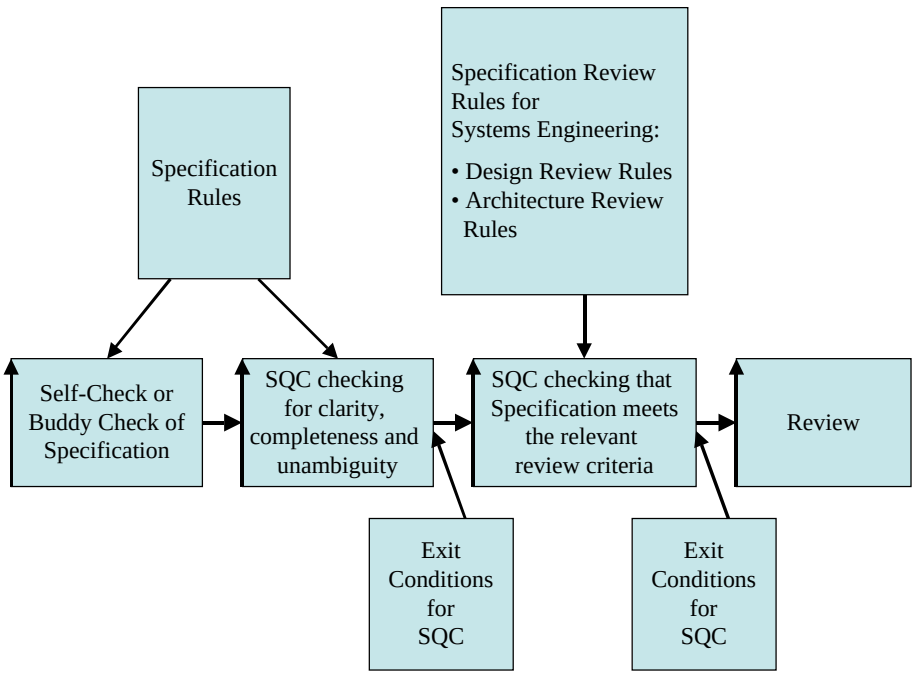

Figure 3: SQC and Review processes. Review should follow successful exit from SQC.

I will present this paper in terms of a set of design review principles.

DR1: If the specifications are unclear, you are not ready to review whether the design is the 'right thing' to do.

DR2: You cannot review design if you are unclear whether a 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.
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 be conducted at an effective rate. Don't overwhelm the reviewer, so they become incapable of spotting problems.
DR8: In order 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 ignorance.

Principles

Principle DR1: If the specifications are unclear, you are 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 specification quality control (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 to 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 themselves 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 sub-contractor in Bangalore').

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).

Most people (for example the writer of the specification and their peers) consistently manage to find between 5 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, if you double the

number of majors found by the checker finding the most majors, then you tend to have 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% to 35% (This is partly due to the time allowed, and partly due to the source information available). For the sake of simplicity, say 33% (a third).

So if 15 majors were found, I'd estimate that there were 15 x 2 x 3 = 90 defects on the single sample page. That 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 our understanding of a whole design, and capable of making our judgments, in a design review, worthless.

The bad news is that this level of majors/page will continue to persist unless management 'changes the game.' The good news is that, if you want to, you can bring down the average level of majors/page by two orders of magnitude, within weeks of doing the right things.

You can bring down the majors/page density by stating the specification rules (such as 'clear' and 'unambiguous'), and then training individual engineers to follow these simple rules. You do 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 1 major/page) is exceeded. Under such conditions, an individual will learn, at a rate of about 50% defect injection reduction per cycle of specify/check/correct. 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 you do this, it works. Others, such as Raytheon Defense Electronics, have reported the same experience working within military software projects (Haley et al. 1995).

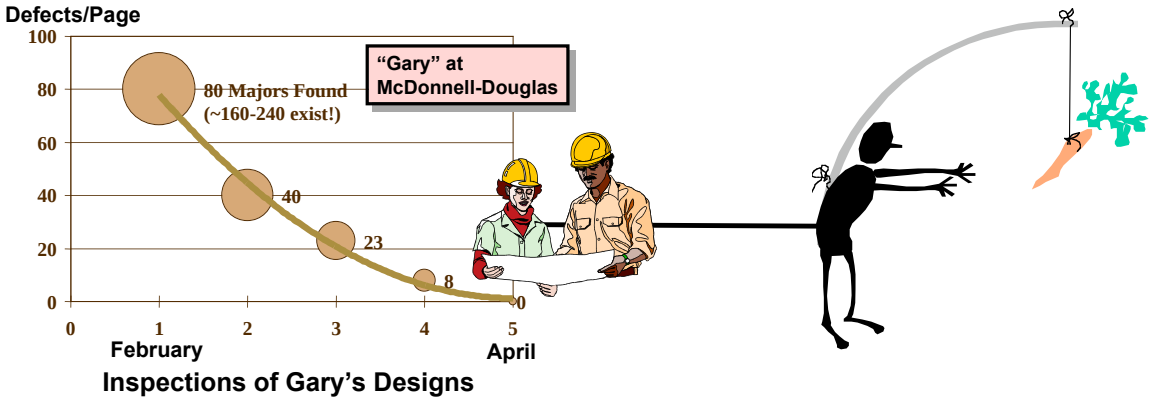

Figure 4: The individual engineer is generally capable of reducing their defect injection density by 50% per cycle of personal learning and feedback (Personal Experience, 1988 and later).

What does all this mean for design reviews? In summary, we need to quality control all specifications and make sure they are not polluted, before we can seriously undertake a design review to look at whether we are doing the 'right thing.'

I suggest that you need at least 2 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

I recommend that you consider setting the numeric level at 'less than 1.0 estimated remaining major defect/page.' Anything less is wasting your review team's time and your project time. What do you do today instead?

Principle DR2: You cannot review design if you are unclear whether a 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 ought to be only in the design specification. (In case you are wondering about this: requirements are the end states needed by stakeholders, irrespective of implementation detail. Design is a decision about how to implement those requirements in practice.)

Of course, mandatory designs (the required design constraints) must also have detailed design specifications. You need to ensure you 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 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).

So, tell me, do your current design documents have integrated, or referenced, the impact estimations for the designs, ready to feed into the design review process?

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 work capacity, qualities, and savings 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 (±%), and
- a credibility rating (on a 0 to 1.0 scale).

This data ideally will be presented on 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.

Figure 5: 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/page would the specification be submitted for further SQC to see if the specification was ready for review (See Figure 6).

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

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 data they are being presented with.

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 in fact adhered to the design specification rules (See Figure 5), there needs to be a check against design review rules in preparation for a design review. Only if the design meets the design review rules should a review proceed.

Examples (Incomplete) of Specification Review Rules for Design

AC1: Are the set of resource costs acceptable in relation to any relevant investment or operational budgets, which exist?

AC2: Are the 'estimated architecture ideas performance impacts' sufficient to justify the resource costs? Are they the best impacts we can get at that cost level?

AC3: Does the suggested architecture, in terms of cost and performance impacts above, fit in with all other know envisaged architecture past, present, future. For example does the specified architecture cause us to exceed any resource budgets (time, money, space effort)? Does it threaten any critical performance level with unknown or known negative side effects?

AC4: Is this architecture specification the arguably best overall option for us? Have other promising options been evaluated? Should they be? Is there any dissent amongst expert architects on this matter?

AC5: Is there a plan for validating the real performance and cost impacts of this architecture in practice, before we commit to it on a larger scale?

Figure 6: Design Specification Review Rules to be used within SQC to check if the design specification is ready for design review.

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 done in Sweden, I saw the signatures of 7 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 3 of the 40,000 pages of logic design. This was of course about one third of what was actually present in the pages (but for managers they were pretty good!). Their divisional director took 30 minutes to personally check the 19 majors – while the 8 of us waited in his office one evening, 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 number 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." Now you know why I am skeptical about review committee approvals!

We had many an interesting discussion on the basis of this finding. The central one was that is they had bothered to do some simple samples early, like 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 about early partial work done sampling here. Don't wait for the entire specification before you do some basic quality control on it – a point we shall return to in this paper towards the end.

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

If an attempt is made to review too great a quantity of 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 of, as in the example above) design specifications are delivered at once to a design review, then the review group will clearly not have time to study, discuss, criticize anything in detail. Reality of risks and problems will be lost. It will not be possible for the review team to learn about their misjudgments. The project or product will probably fail and we will not be clear about the cause of failure.

I suggest that we need to feed only a small volume of ideas into a design review committee:

- in order to give the committee a chance to do its work properly
- in order to make sure that any preparatory work has been done properly

May I suggest one page of design per committee hour as a rough guide?

Principle DR8: In order 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 Evolutionary Project Management (Evo) methods and their generic cousins iterative (*we do cycles*) and incremental (*we cumulate*) 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 (*we learn*!).

Evo means testing the stakeholder effects of a design idea in the field. We are suggesting 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 for implementation of each previous design; and they will learn quickly and realistically, by real experience, how to evaluate designs.

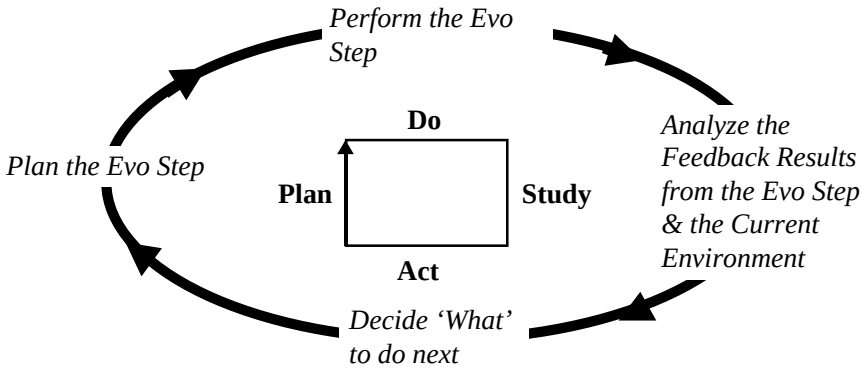

Figure 9: the Deming/Shewart PDSA cycle. The essence of Evolutionary project management and of consequent Evolutionary design processes.

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

Table 1: 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 IE 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. Designs and estimates can then be adjusted and improved from an early stage of the project.

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

Design reviews are about risk analysis, and we need to build a much better foundation in order to reliably carry out that function. This is where Evolutionary Project Management (Evo) methods are helpful. If you do not implement your designs evolutionarily, you might never learn that one particular one of them was your downfall. Even if you do learn which one it was, it is probably too late to do anything about it on this project.

If you 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, not total avoidance of risk!

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 be come obsolete, since reality is going to give designers more reliable advice than any committee.

Conclusions

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

Design reviews should be held throughout the lifetime 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 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, Tom and Dorothy Graham, *Software Inspection*, Addison-Wesley, ISBN 0-201-63181-4, 471 pages, 1993. (13th printing in 2005).

Gilb, Tom, *Competitive Engineering: A Handbook for Systems Engineering, Requirements Engineering,and Software Engineering using Planguage*, Elsevier 2005. Both SQC, Design and Evo have their own chapters in this book.

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

Johnson, Jim, Karen D. Boucher, Kyle Connors, and James Robinson, "Collaborating on Project Success," Software Magazine, February 2001. www.softwaremag.com/L.cfm?Doc=archive/2001feb/CollaborativeMgt.html

Morris, Peter W. G., *The Management of Projects*, London, Thomas Telford, ISBN 0-727-71693-X, 358 pages, 1994.

Taylor, Andrew, "IT projects sink or swim," BCS Review, 2001. http://www.bcs.org.uk/review/2001/articles/itservices/projects.htm

Author Bio

Tom has been an independent consultant, teacher and author, since 1960. He mainly works with multinational clients; helping improve their organizations, and their systems engineering methods.

Tom's latest book is 'Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage' (Summer 2005).

Other books are 'Software Inspection' (with Dorothy Graham, 1993), and 'Principles of Software Engineering Management' (1988). His 'Software Metrics' book (1976, OoP) has been cited as the initial foundation of what is now CMMI Level 4.

Tom's key interests include business metrics, evolutionary delivery, and further development of his planning language, 'Planguage'. He is a member of INCOSE and is an active member of the Norwegian chapter NORSEC. He participates in the INCOSE Requirements Working Group, and the Risk Management Group.

Email: Tom@Gilb.com
URL: http://www.Gilb.com

Version Nov 9 2005