

## Advanced Requirements Specification:

### Quantifying The Qualitative

Tom Gilb,  
Result Planning Limited,  
[Gilb@ACM.org](mailto:Gilb@ACM.org)

**Abstract.** Generally, the quantification of quality is not mastered by current culture. Qualitative requirements can always be specified with quantitative clarity. This paper gives pragmatic ideas on how to achieve this. The main concept is to define a 'scale of measure'.

#### INTRODUCTION

**Current Culture.** There are problems with the general current specification of qualitative requirements. There is insufficient understanding of the importance of these 'quality requirements'. They are crucial because they are our main competitive differentiator for a given functional area, aside from cost. A lack of adequate quality specification tends to limit creativity and prematurely cuts short the system design.

Often quality requirements are placed in categories called 'non-functional requirements' or 'intangibles'. People assume that they can limit themselves to describing qualities at a high-level and qualitatively. We have all seen requirements of the type:

<b>Change:</b> {enhance, improve, reduce, increase}
<b>Quality:</b> {performance, ease of use, maintenance costs, productivity, security, adaptability}

For example:

"achieve substantial increase in our product's ease of use by the customer"
---

For certain purposes, such as motivational management speeches or hiding the precise objectives, such requirement statements can find a use. However, within projects, this non-quantified culture is unacceptable because it is too vague to provide clear direction for activities such as contracting, testing, configuration management and engineering design. The large investments of time, people and money, the need for the high impact results and, the high risks involved demand quantitative specification. Quality requirements must

be defined as clearly as we define the functionality and cost requirements[1]. They must be stated in an unambiguous and testable manner or we risk not getting design and test carried out for them.

**Practical Experience.** Many people are initially reluctant to accept that all qualitative requirements can be quantified. However, my experience (over more than thirty years) is that it can always be achieved. Even in cases, where the quantitative scale of measure has to be indirect, it is far better than none. Once people have seen a few examples of quantitative requirements, they are usually quickly won over.

In fact, there are several companies which already have clear top management policies about quantifying objectives and quality requirements. For example, Ericsson, Hewlett-Packard and IBM. But most companies have not taken a position on the subject. It is important that top management take the initiative and communicate clearly their requirements. The message needs to be backed up by a pervasive company culture and necessary training.

**Approach to Quantitative Specification.** The aim of this paper is to give some practical ideas on how to quantify qualitative requirements. The approach that needs to be taken is rather more than simply introducing numeric values. In overview, the following steps are necessary:

- 1) All the quality concerns for the system have to be identified
- 2) Scales of measure have to be assigned
- 3) The required quality levels have to be stated

The waterfall approach has led to a culture of the specification of the final maximum necessary values as requirements. In a more sophisticated evolutionary result delivery culture, it is useful to differentiate various quality levels for delivery to different stakeholders, under different conditions, at different times. Further, it is desirable to evolve the quality specification throughout the system development life-cycle. Changes in requirements due to alterations in the business environment and feedback from

---

<sup>1</sup> Note, I distinguish several categories of system specification; quality requirements (what quality attributes the system shall possess), functional requirements (what the system shall do), cost requirements (what resources the system shall use), design ideas (what we will do to make the system meet its requirements) and constraints (which create a fence around permitted and not-permitted areas of requirements and design).

## Advanced Requirements Specification: Quantifying The Qualitative

experience with the system in the field need to be captured.

### SCALES OF MEASURE

**Expressing Qualitative Concepts.** Let me show you a set of ideas for describing qualitative requirements. We will start with a simple example about 'weight'.

"Reduce weight" can be expressed (using Planguage, a requirements definition language that I have developed) as follows:

Tag: Weight  
Gist: *To lose enough weight so that I don't feel overweight.*  
Scale: Weight, in relation to average weight for height, age and sex, expressed as percentage against the average weight set at 100%.  
Meter: Observed weighing scale values compared to 'tables of normal weight'.  
Past [Last Year, Me] 130% *"I was 30% over normal weight".*  
Must [End this Calendar Year, Me] 110%.  
Plan [End Next Calendar Year, Me] 100% <- New Year Resolution.

#### Figure 1. Definition of 'Weight'

It is easy to understand what the requirement is just by reading it. Did you notice how much this specification tells you about my requirement to "reduce weight"? Quite a lot of ambiguity is removed. You have a clear idea of the percentage amount of weight to be lost and what the time-scales for the weight loss are. You also know that to avoid failure, a 20% weight loss has to occur by the end of this calendar year and, that success can be declared if I achieve the average weight by the end of the next calendar year. However, it doesn't say where I obtained my 'tables of normal weight' from. It also doesn't tell you my precise weight; an additional scale of measure ought to be specified to fix that!

**Finding Scales of Measure.** If I say 'weight' you think 'Pound, Kilos, Tons, Grams, Ounces'. We have immediate cultural access to certain scales of measure. These scales frequently have internationally acknowledged standard definitions.

However, for most of the quality requirements we will deal with in real life, we cannot think of a scale of measure at all. Try 'adaptability'. Nothing comes readily to mind. We have no training on how to 'develop' scales of measure from scratch. We do not have a handbook to look them up in. In fact, we do

not have a culture that demands that we find scales of measure for them.

So, we give up. We make excuses: 'it is qualitative', 'it is not measurable'; all of which means that we are *unnecessarily defeated* in getting control over these concepts.

This is a problem because our system specifications are full of qualitative concepts, such as 'adaptability'. To give some examples: 'easy to maintain', 'portable', 'secure', 'user-friendly' and 'safe'.

Now, let me introduce an approach to quantifying these concepts. Consider the simplified descriptions in Figure 2. These examples hopefully provide sufficient evidence to show that suitable scales of measures do exist. Further examples of scales of measures can be found in the references.

To find scales of measure for your system's qualities, I suggest the following steps:

- Look in reference documentation to see if you can find any established scales that might be of

**Maintainable:** Scale: The average time taken to repair a defined fault in a defined product.

**Portable:** Scale: The relative defined effort in work hours to move a defined product to a new defined environment compared to the effort involved in constructing from new.

**Secure:** Scale: The % of defined classes of attacks which would fail to penetrate the defined system in defined ways.

**User-Friendly:** Scale: The time needed for defined categories of people to achieve defined tasks.

**Safe:** Scale: The probability that defined dangers will result in defined undesired results.

**Adaptable:** Scale: The work effort in work hours needed to adapt a defined product to include defined new requirements.

*Note: The above are intentionally simple and do not represent recommended scales or what I would do in practice with a real industrial situation. TG*

#### Figure 2. Simple Descriptions for Quality Concepts

use. Maybe there are scales that are close to what you want and you can tailor them.

- Look in the system requirements to try to identify any obvious scales (e.g. "we aim to reduce the amount of paper documents" and "we must reduce staff turnover").
- Think about what you are trying to alter with the system and about how you would measure success.
- If the quality concept is complex, and no suitable



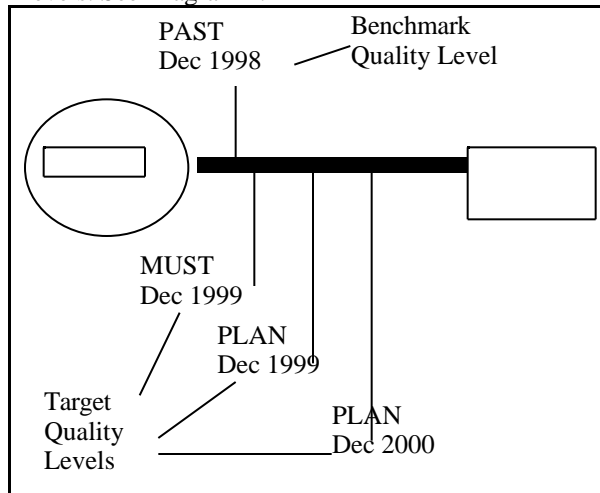
## Advanced Requirements Specification: Quantifying The Qualitative

**Meter:** Statistics from [vehicle repair workshops] gathered at [time intervals] and analyzed according to [time period].

**Meter:** Number of orders for [specific vehicle components] from [parts manufacturers].

The search for alternative Meters should be encouraged with the emphasis on finding cost-effective measuring methods.

**Quality Levels.** The Past, Must and Plan parameters each specify a *quality level* on the defined Scale of measure. Numerous versions of each of these parameters can be needed to capture all the different levels. See Diagram 1.



**Diagram 1: Quality Levels**

The 'level' on the defined Scale is where we get 'really specific' about the requirement. This does not mean we have to know some exact 'correct' value, because 'greater than 20%' or '20% plus/minus 5%' or 'over 20 years old' are also precise in their meaning, without being *unnecessarily exact*. It all depends on the context. The level specification has no meaning, without us also knowing the defined scale of measure and the qualifiers.

**Qualifiers.** Qualifiers are used to document the different times, places/spaces and conditions/events for the given quality levels. Qualifiers are denoted within square brackets [when, where, if]. Many qualifiers are fairly self-explanatory and the Scale parameter can often provide help in understanding them. However, where there is ambiguity or it is not clear, additional information must be provided by defining the qualifiers or adding notes.

For example:

**Tag:** Truancy

**Scale:** Percentage of [SCHOOL-TIME] lost by [Students] at [NAMED-SCHOOL] due to truancy.

**Past**[1997-98, third year students] 10%

**Past**[School Year =1997-98, School-time =whole school-days, Students =third-years at School A] 2%

**Must**[1998-99, third-year students] 9%

**Plan**[1998-99, third-year students] 6%

**Must**[1999-00, fourth-year students] 9%

**Must**[1999-00, third-year students in Class Q] 10%

**Plan**[1999-00, third-year students in Class Q] 8%

**Plan**[1999-00, third-year students, Special Conditions = IF special funds available] 5%

**SCHOOL-TIME:**Defined:whole school days.

**NAMED-SCHOOL:**Defined:School Z.

Note: Class Q is a known difficult class. Special effort needs targeting here.

### TOWARDS SPECIFICATION STANDARDS

The discussion so far in this paper has been about stating quality in a quantitative manner. I want at this point to draw your attention to the importance of using templates and standards to ensure the accuracy and efficiency of your specification process.

**Templates.** I have already mentioned that effective scales of measure should be documented and reused. The same equally applies to proven attribute hierarchies (which can be expressed using Tags and Scales), measuring processes (Meters) and, benchmarks (Pasts, Records and Trends).

**Processes and Rules.** You should capture your best practices by documenting your processes and rules. This will provide clear guidelines of the work expected and standards for quality control purposes. The appendix shows an example of a process and rules for quality quantification.

**Generics.** Whenever documenting templates, processes or rules always consider the possibility of specifying a *generic* version. Generic versions are required when a pattern starts to emerge of repeating similar material with only slight variations (or maybe none). For example, RULES.GR in the appendix is a

**Tag:** Maintainability

**Gist:** Ability to fix faults.

**Scale:** The clock time for

[1. defined Class of faults] from a

[2. defined Start point, Default: first

knowledge of the fault by any system or

## Advanced Requirements Specification: Quantifying The Qualitative

person] to [3. a defined Completion point OR Default: the point where it is in fact correctly fixed, without any negative side effects and this is verified] by [4. defined fixing and testing Process OR Default: any process] using [5. defined Capability of person OR Default: any one who does the job in practice]. <b>Past[P1, P2]</b> Average 24 hours <- Inspection Statistics <b>Plan[P1, P2]</b> 10% of Past. <b>Must[P1, First General Product Release Internationally]</b> 1% of Past <- George X.  <b>DEFINITIONS</b> <b>P1:</b> Defined: {Class = <b>Requirements Fault</b> , Start = Detection Logging in Inspections, Completion = Approved in Inspection Follow up Process, Process = Fixing in Master Specification, Capability = Requirements Author}  <b>P2:</b> Defined: First Release to Field Trial.  <b>First General Product Release Internationally:</b> Defined: the date of access to our product by more than one country, after field trials for paid purposes.  <b>Requirements Fault:</b> Defined: any requirement specification statement judged by Inspection process as a Major defect (i.e. a fault that might cause high downstream (design, test, field) costs). <i>Note: Other terms such as those used in P1 should be defined somewhere.</i>
---

### Figure 3. Generic Definition for Maintainability

generic rule set and, Figure 3 shows a generic scale definition for Maintainability (See also next section).

**Generic Scales Definitions.** Scale definitions can be made generic by embedding 'definition parameters' within them. This enables the necessary detail to be specified by the qualifiers within the individual Past, Must and Plan statements. Notice in Figure 3:

1. The Scale is extremely generic, but it does have automatic defaults, which are reasonable.

2. There are five definition parameters (Class, Start, Completion, Process and Capability) which can be used to define the Scale's meaning.
3. These five definition parameters can be specified in the qualifiers of the Past, Must and Plan parameters. Numerous different definitions can be used to pinpoint the precise qualifying conditions for the differing quality levels.
4. A useful set of these parameters can be defined once (See 'P1') and reused, to simplify and clarify (that the same set are intended). 'Names' or 'labels' can be applied not only to the overall requirement (like 'Maintainability') but to any useful sub-component of the definition (See 'P1' and 'P2').
5. Parameters can be formally defined locally or globally. In the example, a local definition is given for 'First General Product Release Internationally'. But it might be more appropriate to define the term globally, say in a Project or Requirements Glossary. The general signal that a term has a formal definition somewhere else is given by capitalization of the words describing it (e.g. 'P1').

### PRINCIPLES FOR QUALITY QUANTIFICATION

Here is a set of key principles that I think summarize some of the main points of quantifying quality.

#### 0. THE PRINCIPLE OF 'BAD NUMBERS BEAT AMBIGUOUS WORDS'

Poor quantification is more useful than none; at least it can be improved systematically.

#### 1. THE PRINCIPLE OF 'NO INTANGIBLES'

All qualities can be expressed quantitatively, 'qualitative' does not mean 'unmeasurable'.

#### 2. THE PRINCIPLE OF 'MANY SPLENDORED THINGS'

Most quality ideas are usefully broken into several measures of goodness.

#### 3. THE PRINCIPLE OF 'MANY LEVELS'

Uniform quality costs. Understand how your quality levels vary across the system and over time.

#### 4. THE PRINCIPLE OF 'THREATS ARE MEASURABLE'

If lack of quality can destroy your project then you can measure it *sometime*; so the only question is 'how early are you going to measure it?'

#### 5. THE PRINCIPLE OF 'LIMIT TO TEN'

## Advanced Requirements Specification: Quantifying The Qualitative

There is a *practical* limit to the number of facets of quality you can define and control. This is far less than the number of facets that you can *imagine* might be relevant. If you can control your 'top ten' quality concerns then you have probably cracked the problem!

### 6. THE PRINCIPLE OF 'METERS MATTER'

Get your feet on the ground! Practical Meters improve the understanding and application of 'scales of measure'.

### 7. THE PRINCIPLE OF 'HORSES FOR COURSES'

Different Meters will be necessary for different points in time, different events and different places.

### 8. THE PRINCIPLE OF 'BENCHMARKING'

Ignore past history and future trends at your peril!  
They can always act as pointers and *help* define words like "improve" and "reduce".

## CONCLUSION

This paper has presented pragmatic ideas on how to achieve quantification of qualitative requirements. Organizations must insist on quantitative requirements. Only then can there be effective communication between senior management and project teams and only then, will there be any true yardstick for measuring projects' success or failure.

standard of defect-freeness (by default, the standard is less than one estimated remaining Major defect per page).

TAG: **PROCEDURE.QQ**

VERSION:0.1. STATUS:Draft. OWNER:TG.  
DATE:01March1999.

0:RULES: Use applicable rules {RULES.GR,  
RULES.QQ}

NOTE: Realistic quality quantification evolves over the complete system life-cycle. Numerous iterations will be needed, especially at the beginning, to get to a specification with quality levels that all the key people agree to.

STEPS:

1. Build a list of quality attributes from all your process input documents.
2. Check through quality attribute reference documentation to see if it gives you any pointers or ideas that help.
3. Go out and talk to key managers about their ideas. Check out that you have captured all the quality concerns. Keep asking yourself "Why do we want to do this?" until you understand the connection with the top level organizational objectives. This will give you direction about what really matters.
4. Improve your set of quality attributes.
5. Identify SCALES and METERS for each quality attribute. Look up similar scales of measure ideas in books and handbooks, including your own company collection of scales of measure from previous projects. Preferably reuse existing standard SCALES and METERS. Then check the system documentation looking for mentions of scales of measure. If you are still lacking some Scale definitions, try to think up new definitions. If the concept is complex, and no suitable scales seem to cover what you want control

## APPENDIX

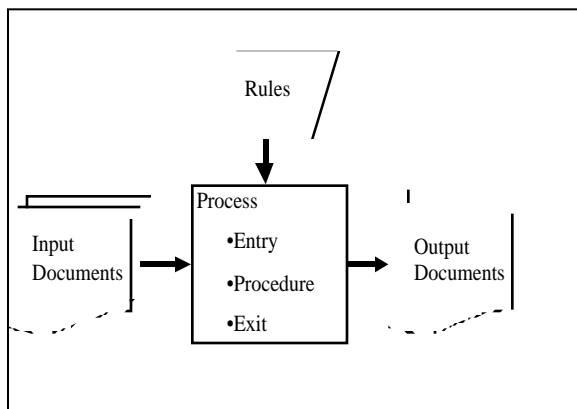


Diagram explains the relationships amongst the terms: processes, entry, procedure, exit and rules.

## EXAMPLE STANDARDS FOR QUALITY QUANTIFICATION

### PROCESS.QQ.

TAG: **ENTRY.QQ**

VERSION:0.1. STATUS:Draft. OWNER:TG.  
DATE:01March1999.

0:COMPLETED: It must be established that there is not already adequate quality quantification. Check if a set of quality attributes with appropriate scales of measure is specified.

1:STANDARDS: All applicable standards (industry standards, project specific rules, etc.) must be identified.

2:INPUTS: All input documents must be identified and available.

3:QUALITY: All input documents must be quality controlled and have exited at a known and acceptable

## Advanced Requirements Specification: Quantifying The Qualitative

over, then consider 'exploding' the concept into sub-concepts until you reach a level of detail where measures become obvious.

6. Use the quality levels stated in the documentation to specify appropriate Benchmarks {RECORD, PAST, TREND} and Targets {MUST, PLAN}.

7. Check you have specified the parameters completely. Use the RULES.

8. Talk through the main aspects of your quality specification with your peers. Iron out the obvious points of misunderstanding.

9. Talk through the main aspects of your quality specification with key managers. They must feel comfortable with the set of quality attributes and the quality levels. Remember you don't need necessarily to use the Planguage format!

10. Perform quality control on the output documentation. If using the Inspection method, calculate the remaining Major defects per page for the exit criteria checking (The rules, RULES.GR and RULES.QQ, provide the Inspection rules).

11. Document any improvements to the process and/or data that have been identified during the quality quantification process and make available to others.

12. Revise these specifications when some design engineering/planning work has been carried out using them. Valuable data (e.g. about the available technology and its costs) is likely to be more available and, the recent practical experience can be captured and considered.

---

### TAG: EXIT.QQ

VERSION:0.1. STATUS:Draft. OWNER:TG.  
DATE:01March1999.

0:QUALITY: The estimated remaining defects per page must meet or be better than the required criteria for exit. By default this is less than one estimated remaining Major defect per page.

---

### TAG: RULES.GR

VERSION:0.1. STATUS:Draft. OWNER:TG.  
DATE:01March1999.

GIST: Generic Rules for Specification of Technical and Management Documentation

0:CLEAR: Statements must be clear and unambiguous to their intended reader.

1:SIMPLE: Statements should be written in their most elementary form.

2:TAG: Statements must have a unique identification tag.

3: UNIQUE: Requirements and design specifications must be made one single time only. They must be

reused by cross reference to their identity tag. Duplication is strongly discouraged.

4:GIST: Complex statements should be summarized by a GIST statement.

5:SOURCE: Statements must contain information about their source.

6:AUTHORITY: Statements should include information about their authority and/or current status level.

7:QUALIFY: When any statement depends on a specific time, place or event being in force then this shall be specified by using [square brackets].

8:FUZZY: When any element of a statement is unclear then it shall be marked, for later clarification, by <fuzzy angles>.

9:COMMENT: Readers must be able to *visually* distinguish critical from not critical specification. Any text which is secondary to a specification, and where no defect could result in a costly problem later, must be written in *italic text* or moved to footnotes. Such text should be headed by a suitable warning, such as *NOTE* or *COMMENT*. Non-commentary specification must be in plain text. However, *italic text* can be used for emphasis of single terms in non-commentary statements.

---

### TAG: RULES.QQ

VERSION:0.1. STATUS:Draft. OWNER:TG.  
DATE:01March1999.

GIST: Rules for Quality Quantification.

0:SCALE: All quality requirements shall define a numeric SCALE fully and unambiguously, or reference such a definition.

1:STDSCALE: The SCALE must wherever possible be derived from a standard SCALE (in named files or referenced sources) and the standard *shall* be source referenced (←) in the specification.

2:SCALENOTE: If the SCALE is not standard or is essentially new, a notification must be sent to the SCALE owner to inform them about this case. 'Note sent to <owner>' will be included as comment to confirm this act.

3:MULTISCALES: Where appropriate, a quality concept will be specified with the aid of *multiple* SCALE definitions, each with their own unique tag, and appropriate set of defining parameters.

4:METER: A practical and economic METER or set of METERS will be specified for each SCALE, to at least sketch the process to be used to determine how we intend to test or measure where we are on the defined SCALE.

5:STDMETER: The METER must wherever possible be derived from a standard METER (held in reference

## Advanced Requirements Specification: Quantifying The Qualitative

sources) and the standard *shall* be source referenced (←) in the specification.

6:METERNOTE: If the METER is not standard or is essentially new, a notification must be sent to the METER owner to inform them about this case. 'Note sent to <owner>' will be included as comment to confirm this act.

7:SUCCESS:The numeric levels needed to meet requirements fully (and so achieve success) must be specified. In other words, one or more [qualifier defined] PLAN level goals must be specified.

8:FAILURE: The minimum numeric levels to *avoid system, political, or economic failure* must be specified. In other words, one or more [qualifier defined] MUST level goals must be specified.

9:BENCHMARK: Reasonable attempt to establish benchmarks {PAST, RECORD, TREND} will be made for our system's past, and for the relevant competition.

10:TIMESERIES: Future-priority requirements (MUST, PLAN) must be stated for both *long* and *short* term.

11:SCOPE: Qualifiers stated in square brackets shall be used to state any useful and necessary information about timing, location or conditions for the requirements levels. The qualifiers must adequately describe the scope of the system.

12:DIFFERENTIATION: Variable target requirement levels for a single scale of measure shall be stated wherever advantageous. Qualifiers must be used to specify the [when, where, if] conditions.

13:NUMSOURCE: The source of all numeric specifications must be stated.

14:NUMAUTHORITY: The authority level and/or the status level of the source data must be stated whenever this is not evident from the source cross-reference.

15:NUMUNCERTAINTY: Numeric values all have a degree of uncertainty. Where there is a significant level of uncertainty, there must be clear indication of the degree of it (plus/minus) and the evidence or reason for it (e.g. "because contract & supplier are not yet determined"). The reader shall *not* be left to guess or remember what is known (or could be known with reasonable inquiry) by the author.

### REFERENCES

Gilb, Tom, *Principles of Software Engineering Management*. Addison Wesley Longman. 1988.

13<sup>th</sup> printing 1997. ISBN 0-201-19246-2.

Gilb, Tom, *Requirements-Driven Management.or Competitive Engineering (new working title Sept.99)*

A draft book manuscript to be found at

<http://www.result-planning.com>

Gilb, Tom, "Quantifying Quality". Paper in *Requireonautics Quarterly*, Newsletter of the Requirements Engineering Specialist Group of the British Computer Society, Issue 12. October 1997. Pages 9-13. An early version of this paper.

### BIOGRAPHY

Tom Gilb has been an independent consultant since 1960. He is author of several books, including "Software Metrics" (1976), "Principles of Software Engineering Management" (1988) and "Software Inspection" (1993).

Tom immigrated to Europe from his birthplace, California in 1956. He spends approximately half his time working in the US and half in Europe.

**Acknowledgements.** Thanks are due to Lindsey Brodie, London, [lindsey@brodie.source.co.uk](mailto:lindsey@brodie.source.co.uk) for helping me edit this paper.