Richard Stonehouse
Software Quality Consultant

Training, advice & audit in
Software Process Improvement
and Quality Management

9 South East Circus Place
EDINBURGH   EH3 6TJ
United Kingdom
+44 (0) 131-225 7020
richard@rstonehouse.co.uk
<URL:http://www.rstonehouse.co.uk>

Richard Stonehouse Software Quality Consultant: Training, advice & audit in Software Process Improvement and Quality Management

Table of Contents

About this Brochure

This brochure contains a printable version of the material in the web site:

`<URL:http://www.rstonehouse.co.uk>`	(visual version) or
`<URL:http://www.rstonehouse.co.uk/aural>`	(aural version).

The main purpose is to provide information about software quality management consultancy in general, and about my consultancy services in particular. Chapter 1 *Software Quality Consultancy* gives an overview of this, and Chapter 8 *Consultancy - Frequently Asked Questions* contains more detailed information.

A second objective is to provide information about software quality management that will be useful to people wishing to set up a quality system, irrespective of whether they intend to use consultancy. This information will be found in Chapters 2 to 7.

The contents of this brochure draw on my experience, details of which are given in Chapter 9 *About Me*. I now practise as an independent Software Quality Management consultant, offering services to UK and European software organisations. These include small-to-medium businesses, departments of engineering companies and projects in large, diversified corporations.

This brochure is also provided with a *Bibliography* of referenced works and further reading, and an *Index* of topics discussed in the text.

Chapter 1. Software Quality Consultancy

1.1. Quicker Results With Consultancy

Would you like to improve your organisation’s quality management?

Do you need to prepare for an external quality audit such as TickIT?

If so, Software Quality Consultancy may help you succeed, more quickly and more surely. This brochure explains how.

1.2. The Skills You Will Need

To achieve these aims, you will need to know what Quality Management is about - its principles, its methods, and standards such as ISO 9000; how software projects work, and how they can go wrong; and how Quality Management applies to software.

You may also benefit by learning from other people’s experience as to what approaches have succeeded for them. Finally, you will need to know how to put all this together, so as to create the right quality system for your business.

1.3. How Consultancy Supplies Them

Consultancy provides the knowledge and skills that you may not have in-house. It does so through advice, training, audit and other services. It saves you time and avoids false starts.

It helps you build your Quality System now; it also gives your people the expertise they will need to manage, maintain and develop the system in the future.

As an outsider, the consultant can provide a fresh view on problems that have seemed stubborn.

Consultancy is flexible. You can choose a package of services to suit your needs - using as much, or as little, as you require.

It is available on tap. You can work at the pace that best fits the resources and time you are able to put into the project.

Chapter 2. Managing for Better Quality

2.1. What is Quality Management?

Increasingly, software users look for suppliers whom they can rely on. They want consistent delivery of products and services that do what they need and expect. This is because business is becoming more and more dependent on software.

Quality Management is the branch of management that gives that assurance. It can win your company a competitive advantage, by building a name for high quality products and services. It can help you avoid project failures and reduce the cost of bug-fixing. It can enable you to gain the ISO 9001/TickIT or other certificate that customers ask for.

Good Quality Management has to be based on a *system.* This ensures consistent and reliable results. It also takes care of much humdrum detail, so your people can apply their innovative skills, imagination and flair to creative work.

The following notes give a brief general account of *Quality Management Systems* (also known as Quality Systems or QMSs). For a discussion of how this relates to software, see *Quality Management in Software Projects.*

2.2. Some Features of a Quality Management System

2.2.1. Customer Focus

"Listen to what your customers are telling you about Quality" *[9]*.

Example: A company receives a lot of complaints about problems that users are having with a new product. The company is proud of the software; it was well designed, carefully coded and thoroughly tested. But the customers find it hard to install because the set-up guide is lacking.

This company should pay attention to its customers. Of course, they want good software, and the company is quite right to follow the best development practice. But the *software* is not the whole *product,* or the only bit that matters. To a user, the documentation is just as vital.

2.2.2. Quality Manager

You will very likely have someone who makes quality their special concern. It may be a part-time job.

This person should act as an advisor, coach, evangelist and monitor. Their role is to help the people who make the product, or provide the service, to achieve good quality in their own work.

2.2.3. Quality Manual

If you are going for ISO 9000, you must have a Quality Manual. This describes your system in terms of standard quality concepts. Quality auditors and other outsiders will use it as their starting point.

2.2.4. Procedures

A procedure is *something that people do.* Often, it need not be written down, depending on your style of organisation. A written procedure is no use unless it is followed in practice.

We define procedures to ensure that routine work is done in a standard way; that different departments can work together and understand each other; and that people need not waste time thinking up new ways to do mundane tasks. They do not give rules for everything; initiative and judgment still count.

Engineering procedures can rely on engineers' professional skill, shared understanding gained by working in teams, and examples of good and bad past projects.

Management procedures define how the work flows through your development or service process, what quality activities you do at each stage, and what records you keep.

Bringing a procedure into use involves a lot of persuasion, listening to feedback and revision in the light of what you hear. If no-one complains, they are probably not using it!

2.2.5. Records

Record keeping is an essential chore. But systems can be designed to ease the burden. Records can also turn out to be useful, even for people - such as many engineers - who do not like keeping them.

Example: A software release procedure might require engineers to fill in a form with details of the software being issued. After early grumbles, they may find it useful to refer back to the details of past releases.

2.2.6. People

Software, more than most types of work, depends on the skill and professionalism of your people.

Trying to manage through rigid rules, that govern everything people do, will not work. Software is too complex, projects often one-off, and unexpected events too common for this.

Example: Companies sometimes have a rule that coding must not start until the detailed software design has been approved by a senior manager. But, often, managers are out of the office when a design needs approval.

It is better to rely on your people to make their own judgments and take responsibility for them. In the example, the manager's approval is quite likely based on the report of a technical review. Why not let the review leader give the approval?

2.2.7. Management

Senior managers make the organisation's policy and set its tone. They must ensure there is a defined quality system.

Managers lead by example, mainly through the decisions they make. They need to give a consistent message.

Example: The Managing Director authorises a pre-release of an incompletely tested product, at the request of the customer. But the company's quality policy says that software must be fully tested before release. Staff see the pre-release as going against this.

You need to make sure that staff understand the reasons for this decision. Quite possibly it can be justified on quality grounds - see *Customer Focus* - though there are some hard questions to be asked. But you need to explain this to staff.

2.2.8. When things go wrong

Inevitably, things go badly wrong from time to time - no Quality System is perfect. We need to correct them, but also to learn from the mistakes.

Example: Consider again the example under *Customer Focus.* Here, the company first needs to fix the immediate problem by writing a set-up guide. But the important thing is to put the process right, so that the necessary documentation is produced for *every* product. Otherwise the problem is bound to recur.

This approach, of analysing and correcting the root causes of quality failures, is one of the main ways in which we can improve quality. For further discussion, see *How Software Projects Go Wrong.*

2.2.9. Performance Improvement

You need not wait for things to go wrong before taking steps to improve quality. You can measure the quality of the company's products, and how effective its processes are, at any time.

You should collect and analyse such data in a systematic way. The programme needs well-defined goals, rather than amassing huge amounts of data at random.

2.3. Setting Up the System

2.3.1. Where Are We Now?

First, to decide on the best approach, you should assess your current system:

- If it has most of the features described, you are probably well on top of Quality. You may wish to look at the pages: *How Software Projects Go Wrong*; *Quality Management System Standards*; and *The Quality Audit*, for further ideas on how to improve it.
- If you are not sure how well it measures up, you could review it against one of the standards described in *Quality Management System Standards*, or get a consultant to review it for you. Then you will be able to decide how best to move forward.
- If it is not much like what we have described, there is work to do! The key is to do a bit at a time. Aim to sort out one area thoroughly, and get your solution working and delivering real benefits, before you tackle another.

2.3.2. Getting Started

Next, you will need access to knowledge and skills in Quality Management, Software and how they fit together.

One way is to teach yourself. This is excellent, but takes time. These pages, with some of the works cited in the *Bibliography*, may give you a start.

Another is to employ an in-house expert. The risk is that they could then be the only person who knows how the system works. You need to ensure this knowledge is shared.

You could also engage a consultant. You should adopt one of the above two approaches as well, because consultancy is no substitute for in-house expertise; but it can help you get up to speed more quickly.

Chapter 3. The Quality Audit

3.1. What is an Audit?

3.1.1. Types of Audit

An audit means someone coming in to your organisation and looking at how it works, to see whether it meets requirements.

Here, we are concerned mainly with an audit by a *certification body*. They will audit you if you ask them to certify you to a standard such as ISO 9001/TickIT.

Another kind of audit is that done by a *customer*. They may want to satisfy themselves that your company is capable of fulfilling a contract. Such an audit is often similar to that described here.

Yet another kind is an *internal* audit. This is done by your own people, to check the Quality System is working. It differs in detail from what these notes describe, but many of the ideas are the same.

3.1.2. What the Auditors are Looking For

You need to be able to show how your procedures fulfil the requirements. Depending on the type of audit, these may be the requirements of a published standard, of a contract or of your own Quality System. You also need to have records that show compliance in practice.

If you can do these two things, the audit should not cause too much anxiety.

A consultant can advise as to whether you are ready for an audit. He or she can also give detailed advice on the sort of things the auditors are likely to pick up.

3.2. What Happens in an Audit

3.2.1. Before the Audit

The auditors will want your Quality System to have been in use for some time. Typically, this has to be at least 3 to 6 months. You will need records to show that the system is being used and is effective.

Ideally, they would like to see some projects that have gone through the whole system from start to finish. If you have only been operating it for a few months, this will not be possible - some of the projects now in progress will have started under previous systems. But you may still be able to put up a good enough case to satisfy the auditors.

3.2.2. Documentation Review

The auditors need to be satisfied that your documented procedures comply with the standard, contract etc. They used to do this part of the audit off-site, beforehand, though now it is often rolled in with the audit visit.

Document requirements are less stringent than they used to be. Auditors do *not* want a lot of written procedures. Especially, they do not want them to exist on paper only, with no relation to reality!

3.2.3. During the Visit

The purpose of the audit visit is to check that you are following the procedures you say you are, in your day-to-day work.

The auditors interview a cross-section of staff. Interviews generally take place where the work is done, so the auditors can see it happening. They will also call for documents and records. Make sure your people know the procedures they are meant to be following and where to find things.

During their work, the auditors will note any discrepancy they find as a *non-compliance.* If they find a major non-compliance that would result in your failing the audit, they will tell you at once and ask whether you want them to go on with it.

The auditors will be keen to see that you have an effective system of internal audits and that you correct any non-compliances so found. This is important, because the auditors themselves only review a small sample of your work. Be prepared to disclose problems you have found and fixed - these should not count against you.

At the end of the visit, the auditors usually hold a closing meeting. At this, they present a preliminary report of their findings.

3.2.4. After the Audit

After the visit, the auditors write up their report and send it to you. The most important part will be the Audit Findings, a classified list of non-compliances. If an essential part of your quality system is missing or not implemented, this will be a *major* non-compliance and will result in failure. Other non-compliances will be classed as *minor.*

To complete the process, you have to correct the minor non-compliances. When you have satisfied the auditors that you have done so, you get your certificate.

Chapter 4. Quality Management System Standards

4.1. Standards and Certificates

Standards such as ISO 9001, the CMM or the EFQM define what is widely considered good Quality Management practice. They provide a yardstick against which Quality Systems can be assessed.

This means that, if you are certified to such a standard, it can back up your claim to be a quality company. It may help you to win business in markets, such as government, where certificates carry weight. It can also act as a concrete target for your quality improvement efforts.

To do this, you will need to have your company's procedures and practice audited by a Certification Body. If you follow good Quality practices, you should have few real problems in this, though it will take some time and effort. Do what makes business and quality sense, and you should be able to justify this in terms of the relevant standard.

The following notes refer mainly to the current (year 2000) edition of the ISO 9001 standard, as applied to software through the TickIT scheme. Brief reference is also made to other standards.

4.2. ISO 9001 and Software Quality

4.2.1. Help or Hindrance?

The best-known Quality Management standard in the UK is ISO 9001 *[11]*. Software people often refer to it as *TickIT*, because the TickIT scheme provides the route to ISO 9001 for software organisations.

In the author's opinion, going for ISO 9001 certification can be worth while, though it is by no means the only route to quality.

There are differing views. Supporters of ISO 9001 say it enshrines best practice, assures customers of the quality of what they are buying and brings demonstrable business benefits. Critics say it fosters an out-of-date "command and control" attitude and the failed "inspection-based" approach to quality *[17]*.

The author considers both these views over-stated. In any case, if your customers demand ISO 9001, you may *have* to do it - and should seek to obtain the maximum benefit from the exercise.

4.2.2. The Paper Juggernaut

Some Quality Systems are far too heavy. People may write too many, too detailed procedures, which no-one will read. They may make inflexible rules that will just be ignored. They may demand authorisation signatures that are impossible to obtain.

It is a mistake to think that ISO 9001 requires such a system. It does require some documentation and records to show you are complying with the standard. But you need much less than many people realise. And once you have made the effort, it may help you in your day-to-day work.

4.2.3. ISO 9001 in a Nutshell

ISO 9001 may seem complex and obscure to the non-quality specialist; but it can be reduced to just three main ideas:

1. You must be committed to supplying the products or services that you promised the customer, in a fit state. For example, you cannot ship an unfinished product as a ’final’ release - unless, of course, the customer has agreed to accept it in that state.
2. You must have a system that enables you to live up to that commitment. For example, before you submit a tender to develop and deliver a product by a specified date, you must be sure that this is feasible - taking account of technology, risks, work estimates, resources, dependencies etc.
3. You must monitor your system’s operation and continually enhance it.

Part of the problem with ISO 9001 is that it is oriented towards manufacturing, and software is a *design* process. There are guidelines *[12]* that interpret ISO 9001 for software. In the U.K., the TickIT scheme and especially the *TickIT Guide [8]* provide detailed guidance on applying ISO 9001. Consultancy may also be useful.

4.3. Other Standards

Other standards sometimes asked for include the Capability Maturity Model or CMM *[10]*, *[4]*. This is popular in the USA and with U.S. based organisations. It is specific to software. The company is assessed as being at one of five *Capability Maturity Levels.*

The EFQM Business Excellence Model is a model for the whole business, not just the software part. It uses self-assessment.

Various industry sectors have their own sector-specific standards, and may ask software suppliers to comply with them. Often, the standards are closely based on ISO 9001.

Chapter 5. How Software Projects Go Wrong

5.1. The Need for Change

One way we can improve the Quality Management System is by learning from those occasions when it goes wrong. No system is ever completely problem-free, unless - perhaps - it is obsolete. But we want to avoid such problems so far as possible; they cost money, waste time and incur risk.

This is an opportunity to gain an advantage over the competition. Over-optimism, crises, timescale slips, poor field performance and customer disappointment are endemic in the software industry. Brooks wrote about the "Tar Pit" *[3]* in 1975; high profile I.T. based projects still go wrong; and there is a perception that "Not all computing projects fail - only most of them" *[5], [16]*.

The following notes[1] illustrate a few of the problems that can occur in software projects. Some of them may be familiar, or suggest things that have occurred in your experience. If not, then either you are very lucky, or you already have an excellent Quality System!

5.2. Some Software Quality Problems

5.2.1. Late Delivery

Projects may run late for many reasons. The initial estimates may have been too low. There may be a lack of suitable resources. The original feature list may have been too ambitious.

One of the most common reasons is feature creep.

Example: During a large, complex software project, the customer asks for many extra features. The company agrees to all of them. By the time the project is a year behind schedule, both company and customer are worried.

This is bad for everyone. The customer might have preferred the product on time, without the enhancements, had the impact been made clear. The company is out of pocket, as it most likely cannot charge extra for the work.

A good Quality System would have controlled these changes. This does not mean it would prevent or inhibit change. But it would ensure that the full impact was understood and allowed for. One way of doing this is through *Rapid Application Development [18]*.

5.2.2. Keeping the Customers in the Dark

Customers expect to be told what is going on. This is most important in times of crisis. But companies too often neglect it.

Example: A web hosting company suffers a major system crash. Staff work round the clock to reconstruct customers' data. Customers keep ringing up to ask "what's going on?", "when's it going to be back up?" and "why did this happen?" The company's staff get fed up, and either stop answering the phone or give very offhand and unhelpful replies.

The customers' concerns are reasonable and have to be met. But the staff's reluctance to be distracted from the job in hand is also easy to understand. The problem is that no-one has thought to provide a better method of keeping customers informed. Such planning should be part of the Quality System.

5.2.3. What software is this?

Mix-ups over which version of a software item is which, what an item's status is, or which items go with which others, are common. The results can be embarrassing or worse.

Example: After a company issues a new product version, an old fault reappears. It was fixed in the previous version, but the fix was not included in the official copy of the master source.

To prevent this sort of error, you need Configuration Management *[1]*. This should be part of the software Quality System.

5.2.4. Let Down by Others

If the company's products or services fall short, the company takes the blame - even when it may seem to be someone else's fault.

Example: The customer holds the company to blame for late delivery of a product. This occurred because a supplier, to whom the company had sub-contracted some specialist work, went out of business.

It does not matter if the problems are the fault of a supplier, sub-contractor, partner or even the customer. The company made the arrangements and has to make sure that they work.

To do this, you need to identify, analyse and control *dependencies* and *risks.* This is an essential part of project planning and control. You also need reviews to check that the plans were realistic and the control effective. All this should be part of your Quality System.

5.2.5. The Product that Should Not have been Released

There is always pressure to get products out of the door, so as to earn revenue and free staff for other work; but shipping them *too* soon can have quite the opposite effect.

Example: A product is released, although it is still very unstable. The result is a deluge of customer complaints. Engineers are sent on site to try to fix the problems, so work on other developments stops. This sets back potential revenue, which adds to the financial shortfall caused by the support costs. And what happens when those other products are due for release, but aren't ready - will they be sent out in an unfinished state, too?

Clearly, this product should not have been released when it still needed so much more work, however strong the pressure. But we need to look deeper. Were early signs of trouble overlooked? Was the true scale of the project ever really understood? Why did the problems not come to light?

Part of the solution is better project planning and control. This is the job of line management. But some of the issues to be faced may be quite hard. You need a Quality System, and an independent Quality Manager, to make sure they get attention.

5.3. Savings Through Quality Management

5.3.1. The Cost of Poor Quality

Problems such as these cost money. How much time and effort are spent on testing software, correcting faults, re-testing and so on *[2]*? This work need only be done because the software contains faults. In other words it is time, effort and money wasted.

In the long term, they may even threaten the security of the company. For a large software company, failure of a major project can have a severe impact. For a small company, it could be fatal.

5.3.2. How Quality Management Helps

We have seen how elements of a Quality Management System can help with each of these problems.

But the essential thing about a Quality System is that it is a *system*. Only by having such a system do you ensure that what is needed actually gets done. The quality company does things by design, rather than forever being driven by accidents or crises. There is a culture of conscious decision-making based on facts.

Better Quality means fewer faults. This will save rework, support and maintenance - and so reduce costs. As Crosby says, "Quality is Free" *[6]*.

Notes

1. These examples draw on real-life events, but have been generalised and adapted for these notes.

Chapter 6. Quality Management in Software Projects

6.1. Software and Quality

The Quality software company is a good place to work. Quality Management can make the difference between the success and failure of a project; it can bring managed innovation in place of chaos; and it can make for a happy workplace instead of a dead-end.

To achieve this, we need to find a way of applying Quality Management to software. Much Quality Management *practice* came from areas such as manufacturing. It does not always carry over well to the software industry. But the *principles* still apply; they just need re-interpreting.

The key is to understand both Software and Quality Management, and how they relate to one another. Books, such as those listed in the *Bibliography,* and training courses may be useful. Or a consultant can provide training to suit your needs.

6.2. Quality for Software Professionals

6.2.1. Developing a Quality System from Scratch

If you are a software person, you may at first find Quality Management strange.

Example: Quality Management documents often stress the importance of calibration of test equipment. What has this to do with software quality?

To answer questions like this, you need to know the concepts and motives that underlie Quality Management, so you can translate them into software terms. (In this case, the software analogue might be control of test programs and data.)

6.2.2. Adapting a Pragmatic Quality System

If you are already using a system that works well for you, you may want to be able to show that it complies with good Quality Management practice.

Example: Your company wants to get into the defence systems market and so needs ISO 9001.

You will not want to clutter this good system with irrelevant jargon. If it does what you need, you can almost certainly justify it in terms of ISO 9001.

The advice that you may receive, to add a lot of boiler-plate on things like purchasing, document control and records, is most likely wrong. The only case where you might want to do this, is if you have a contract that demands a traditional style of Quality System documentation.

6.3. Software for Quality Professionals

6.3.1. Extending the Quality System to Software

If software is just a part of your business, you may already have a company-wide Quality System and wish to bring the software within its scope.

Example 1: You supply electronic systems that have a software component.

Example 2: Your in-house I.T. services group supports CAD software for the design department.

If you do not have a background in software, the way in which the software department operates may seem strange and perhaps a little undisciplined. But the differences are not really so great.

6.3.2. Software is Different - But Not Very

The first problem will be that the software people speak a different language.

Example: What Quality people speak of as *design*, a software engineer will call *development.* Design means something different to them.

Apart from language, there are some real differences.

Unlike hardware products, software is intangible - you cannot tell whether it is good or bad by looking at it. At the same time, it is quick and deceptively easy to change - so ill-considered changes sometimes cause havoc.

It is complex - the software part of your Quality System may turn out bigger than the rest of it put together.

Most important of all, software is a *design product.* Software development is a *design process* - so every project is a one-off. Also, software is not usually subject to manufacturing faults or to wear and tear, but only to design faults.

Because of these differences, you will need to gain an understanding of how software projects work. There is no need to become a programmer, but you do need to know the language, the process and the management approach.

Chapter 7. The Right System for Your Business

7.1. What is the Right Quality System?

The right Quality Management System is the one that helps you handle the quality issues that arise in your particular business. Each business has its own, unique needs; no one system will suit all.

By contrast, an *unsuitable* system can do harm. It may act as a brake on development. It could get in the way of doing what is right for your customers. It is likely to demotivate staff and management.

These notes give a few hints on designing your Quality System so that it works *for* you, and does not get in the way. A consultant, with experience of many types of organisation, can give further advice.

7.2. How the Type of Business Affects the Quality System

7.2.1. Custom and Customised Software Products

Here, the Quality System needs to focus on your interactions with the customer.

These occur throughout the project. At the start, you write the specification and agree it with the customer. During the project, the customer may ask for changes, and you may produce interim releases for evaluation. At the end, you commission the product and the customer accepts it. After the end of the project, you provide support and maintenance to the customer.

The Quality System needs to provide a framework for these. For example, you will have a standard for product specifications, to make sure that the right things go in and that the extent of your commitment is defined.

7.2.2. Packaged Software Products

Here, internal controls take the place of the actual customer. Product ideas come from inside the company. Market needs come via the marketing or sales department.

The Quality System needs to ensure that these controls work. Interested parties must be able to review and comment on the product as it evolves. You will need procedures for in-house testing and validation, and for field beta testing.

7.2.3. Software Components

Here, software is ancillary to the main business.

Example: A company makes electronic devices. These include control software, which provides the user interface and analyses the data that the devices capture. The software is developed at the same time as the hardware, and the interface between them changes constantly.

The Quality System should focus on the way hardware and software teams work together. You must rigorously control hardware and software configurations. The plan should provide handover points. You will need several stages of testing.

7.2.4. Software-related Services

Here, the service is often performed at the point of delivery, rather than in-house.

Example: A company installs, trouble-shoots and supports computer networks. Its engineers spend most of their time on customer sites.

The quality of service depends on the attitudes, knowledge and skills of the people who do the work. So, the Quality System has to cover training, supervision and control. Planning is important, especially for a standard service such as a training course.

7.3. How the Type of Organisation Affects the Quality System

7.3.1. Large Organisation

A large company may have a problem in ensuring that all its people work to the same standards and can communicate.

Example: A company has offices in several countries. It employs a few hundred engineers, who come from various cultural backgrounds and may speak different languages. They communicate by telephone, e-mail etc, which can cause mix-ups.

The Quality System is likely to be quite formal. It may need to provide common standards for such detailed things as code layout, so that people can understand one another’s work. It will be heavily documented.

To cope with this, the company may have specialist groups for quality assurance, quality control, measurement etc. But it is big enough to afford them.

7.3.2. Small Company

A small organisation cannot afford the kind of Quality System that a large company would use - but does not need to.

Example: A company has half a dozen engineers working in the same office. All of them know what everyone else is doing, and discuss their technical problems with one another.

The paperwork, rules and procedures of the large organisation would stop this company dead in its tracks. There would not be time to write and maintain such a system, and it would stifle the innovative spirit.

Such a company would use a very lightweight quality system. This could rely on the fact that the engineers are highly skilled, know one another's methods and talk to one another.

7.3.3. Organisation with Existing Practices

Your people will have developed many good practices for themselves. These may be unwritten, incomplete, perhaps with some flaws, but nevertheless *they work*, and are methods that staff are comfortable with and feel they own.

Example: A software group has an ad-hoc configuration management system, which uses home-made tools and a set of working conventions.

There is *no point* in replacing this by a bought-in system, perhaps having more features, unless such a system will provide clear benefits that justify the cost and upheaval.

Getting people to use the Quality System, and getting it to work properly, is the hardest part of setting it up; make the job easier by building on existing practice wherever possible.

7.4. Is There a Short Cut?

Designing the system that is exactly right for your business will take time and effort, from people who understand the business and know how it works. This means engineers and managers, whose time is scarce.

Companies sometimes try to short-cut this, by bringing in a ready-made system from elsewhere.

This is a false economy. As we have seen, someone else's system is most unlikely to be right for your business. At best, the quality manuals will sit on the shelf gathering dust. At worst, the system may do actual harm.

7.5. The Benefits

The right Quality System will focus attention on those aspects of Quality that make a difference to the business, through better products and services and fewer failures. It will be relevant, credible and understandable to staff, so that they will accept it and follow it. It will save time by making routine tasks into procedures, some of which you may be able to automate. It will leave scope for initiative, to the degree that the professional skills of your people warrant. And, finally, it will be easy to alter when required - as you own it and understand it, you can adapt it to suit your needs whenever they change.

Chapter 8. Consultancy - Frequently Asked Questions

1. Is it expensive?

It needn’t be. You can use as much, or as little, consultancy as you need. For a small to medium business, it typically works out at between one day a week and one day a month. Consultancy rates may seem high, compared with salaries - but the difference is not so great when you allow for staff overhead costs. Also, you have the advantage of being able to turn the consultancy on or off as required, so you pay only for what you use.

2. How long will it take?

This depends on factors such as the size of your organisation and the state of its current practices. If you are going for certification, the *minimum* is about 6 months, as the auditors will want to see a track record of your Quality System in use. But it would only be as quick as this if your procedures were already in very good shape. To set up a Quality System from scratch would take a good deal longer, and it might be wise to plan the work as a series of distinct phases.

3. How will I know what it’s going to cost and how long it will take?

The work to set up a Quality System needs to be planned in much the same way as a software project. One approach is to have the consultant conduct an initial audit of your organisation’s current systems, then report back to you and agree a joint plan.

4. Will the project be completed on time?

This depends on thorough planning in advance, as discussed above, and effective control during the project. Consultancy will help to keep it on track. The most common cause of delay is that people, whose input is needed, have been called away to do other things. The consultant should spot these problems and report them immediately, while there is still time to recover any slippage. The consultant’s well-defined role, independence and direct reporting line help here.

5. What will I get for the money - what will be the results of the project?

The project will produce reports by the consultant and Quality System documents. It will provide your people with knowledge and skills, either through formal training courses or on-the-job. Most important, it will deliver your Quality System, in use in your organisation.

The consultant cannot guarantee that you will, for example, gain an ISO 9001 certificate. This depends so much on your own organisation. But the consultant can advise you as to whether your organisation is ready for the audit.

6. Will the consultant become indispensable?

No. A good consultant will aim to transfer knowledge and skills to your organisation, by "doing and showing" - there should be no attempt to hoard knowledge. You have full ownership and control of your own system, so you can continue the development yourself or get someone else to do it. Of course, most consultants will be delighted to provide follow-up help if you ask for it - but you do not have to do so.

7. Will the consultancy be provided by experienced people?

You need to discuss this with your prospective consultant. The author’s policy is to provide all services in person.

8. What topics can the consultant advise on?

You should expect a software quality consultant to be able to advise on all the Quality Management topics covered in these pages. Some consultants may have experience in other fields as well. You should also be aware that the emphasis of different consultants may vary; for example some could be people-oriented and others more systems-oriented.

9. If I need help at short notice, can I get it?

You should be able to. Most consultants will try to satisfy clients’ emergency needs, subject to other commitments that they have to honour. You should discuss this with your prospective consultant.

10. Can I out-source the Quality System development to the consultant?

Yes, to some extent. The consultant can work at your premises, more or less as an extra staff member, over a period. This approach will still require a good deal of your people’s time, because the Quality System will need to be based on their input. It is usually less good than doing the work yourself, but can produce a usable Quality System.

11. Can I suspend the consultancy if my people have to go and do other work?

You can, though you should be aware of the possible risks and take action to minimise them. It is wise to make a firm plan for resuming and completing the project; otherwise, experience shows, it may be put off indefinitely. You will need to allow extra time for getting back up to speed after restarting the project. You should also make clear to staff what is happening; otherwise, they may think the company is dropping its commitment to quality.

12. What happens during a consultancy visit?

The consultant works mainly with your quality person. They may often call other members of staff in for meetings as well.

One of the main purposes is to gather information about how the company’s process works, or how people feel it should work. Draft procedures may be sketched out. Work done since the last meeting may be reviewed, and there may be reviews of the overall Quality System from time to time, as it takes shape. If participants have prepared thoroughly, it is possible to get through a lot of work per session.

Another use of these visits is for training. This may comprise a formal course for a group, or less formal one-to-one training.

13. What happens between consultancy visits?

You are likely to find that you have plenty of homework to do for the next meeting. For example, Quality System documents that were agreed in outline will need to be written up. The consultant can review these and send you comments, to be discussed at the next meeting.

14. Can I appoint a junior member of staff to work with the consultant, and avoid involving engineers or managers?

This will not work. Such a person is unlikely to grasp how the organisation as a whole functions. They will lack the authority to demand help from the people who know the answers. Also, they will probably not be able to get changes agreed and accepted. You need a person of sufficient seniority, who is able to call on engineers and managers as required.

15. Will the consultant tear up the work we have done for ourselves?

No; a good consultant will recognise the value of what you have done and the commitment of your people to it. It should form a solid foundation for a full Quality System.

16. Will the consultant insist on us doing things in a way that does not suit our business?

A good consultant, who knows the software industry, will not make this mistake - see *The Right System for Your Business*. Where this has happened in the past, it has usually been due to people who did not understand how software projects work. Such people may place too much emphasis on traditional quality management, instead of the things that matter for software.

Chapter 9. About Me

9.1. Experience

My consultancy draws on a career in software. This gave a practical grounding in how projects work and how they can go wrong, and included board-level responsibility in a start-up software company - contributing to its successful sale.

Software Quality Management experience covers testing and quality assurance, including ISO 9000 and TickIT.

9.2. General Information

9.2.1. Date of birth

- 9th July 1946.

9.2.2. Education

- Ashby de la Zouch Boys' Grammar School.
- Trinity College, Cambridge, reading Mathematics.

9.2.3. Qualifications

- BA degree (class 2.1 honours) in Mathematics.
- BS5750 Internal Audit Certificate.
- TickIT Auditors' Training Course (continuous assessment and written examination).

9.2.4. Assurance of Professionalism

My qualifications and experience are recognised, and my conduct is regulated, through professional affiliations in:

- Quality Management: Institute of Quality Assurance (http://www.iqa.org/) member,
- Software: Chartered Engineer (http://www.engc.org.uk/) and British Computer Society (http://www.bcs.org.uk/) member, and
- Scottish software business community: Member of ScotlandIS (http://www.scotlandis.com/) - the trade body representing the digital industries in Scotland.

9.3. Career History

9.3.1. 1992 - Present: Software Quality Consultant

Consultancy in software quality management for computer software companies. Advice to clients on:

- Application of formal quality assurance (including ISO 9000/ TickIT) to computer software development.
- Documentation, Quality Manuals and Procedures - covering structure, format and essential content requirements.
- Specific Quality Management techniques such as Requirements Specification, Reviews and Configuration Management.

Quality Audit of Client’s Process and System against the requirements of ISO 9000 or other client-specified target:

- Reporting strengths and weaknesses of client’s Quality System.
- Recommendations on audit results and joint development (with client) of quality improvement plans.

Advice to client management on resolution of specific quality problems affecting processes or products, as requested by client.

Preparation and delivery of presentations and courses (including *Quality and ISO 9001* overview and Internal Audit training).

Keeping up with the state of the art by background reading in quality management and software engineering, membership of professional bodies and by attending courses in:

- BS 5750 Internal Audit,
- TickIT Audit,
- Visual Basic Programming,
- Introduction to Java and
- Introduction to DSDM.

Carried out survey of *Quality Management in the Scottish Software Industry* and published report (at own initiative).

Membership of *Quality Group* of *Scottish Software Federation*, helping organise events to improve Quality awareness in the software industry.

9.3.2. 1983 - 1992: Office Workstations Limited, Edinburgh - Designers, developers and suppliers of software products using "Hypermedia" technology

A founder member of the company, Director (until sale) and Secretary.

Quality Assurance Manager

- Introduced a basic quality system including planning, specification, implementing, testing, release, maintenance and support procedures.
- Planned and launched a full-scale quality programme. Established a Quality Committee. Involved management and staff. Supervised procedure writing. Wrote and published Quality Manual.
- Guided company through successful external quality audit.
- Attended formal training courses, read standard works on TQM and Software Quality, and pooled knowledge with other companies.

Test Manager

- Managed system testing of a new product. Planned and specified tests, organised staff, established procedures and advised on release.
- Jointly managed system testing of another major product. Introduced innovative methods to obtain better return on investment in testing.

Company Secretary

- Managed company’s contracts, liaising with sales and company lawyers. Prepared standard contracts. Took part in negotiation. Handled contractual product liability and warranty issues.

- Assisted in sale of company to Matsushita Electric Industrial Co. Ltd., Japan (owners of Panasonic and Technics brand names). Produced detailed response to purchaser's "due diligence" investigation. Liaised with company's lawyers over warranties, contract terms and documentation.
- Planned and delivered a training course for staff of a major customer.

9.3.3. 1971 - 1983: ICL System Programming Division - Designers, developers and suppliers of computer hardware and associated software

Software Technology Manager (PERQ Workstation project)

Chief Designer (2966 DME/S4 emulation)

Designer/Implementer (2960 DME/S4 architecture)

Designer (Variable Computer System)

9.3.4. 1967 - 1971: English Electric Computers, Bureau Division - Designers and developers of computer software application packages, and providers of computer bureau services

Team Leader (Interact 75 multi-access system)

Appendix A. What's New in ISO 9000:2000?

A.1. The Effect of the Changes

Software developers and service companies will need to make significant changes to their Quality Management Systems in order to comply with the latest revision of the ISO 9000 standard.

Organisations seeking certification for the first time will have to comply with the new standard. Those previously certified to the 1994 standards will be required to upgrade their Quality Management Systems over a period.

ISO 9000:2000 is a radical revision of the 1994 edition. It has relaxed some former requirements, introduced new requirements, and restructured and reworded the documents to make them easier to understand.

The impact of the changes on software companies may be less than for some other types of organisation. Many of the new requirements are for things that they would have been doing anyway. However, they will need to pay attention to:

A.2. Process Approach

The new standard requires a *process approach* to Quality Management. Your Quality Manual must define:

- the processes that make up your system,
- the sequence in which they take place, and
- the interactions between them (that is, the data and other information that flow from process to process).

See ISO 9001:2000 sections 0.2, 4.1 and 4.2.2 c).

A.3. Customer Requirements

You must now go and find out your customer's needs - not just rely on the customer to tell you. This includes implied needs and statutory and regulatory requirements, as well as those needs the customer actually states.

This will be especially important when, as is often the case, the customer lacks in-depth knowledge of software matters. The *caveat emptor* approach will no longer do.

There is also a need to communicate effectively with the customer on various matters. This will be particularly relevant to software projects, where the requirements tend to evolve throughout the life of the project.

See ISO 9001:2000 sections 7.2.1 and 7.2.3

A.4. Measurements or "Metrics" and Continual Improvement

Software companies often treated the measurement clauses of ISO 9000:1994 as optional, or paid lip service to them. Under the new standard, you must take measurement seriously and apply it as a tool for continual improvement.

Measurements to be taken include:

- Product quality - as evidence of fitness for release (approval of which has to be recorded).
- Process measurements - to be used for improving the processes.
- Customer satisfaction - how to measure this will need careful thought.
- Corrective actions taken to correct system, process or product faults - to be evaluated in order to ensure that the faults are prevented permanently.

Measurements not only have to be made, they also have to be analysed and applied for a variety of purposes stated in the standard.

See ISO 9001:2000 sections 8.1, 8.2.1, 8.2.3, 8.2.4, 8.5.1 and 8.5.2 c)

A.5. Top Management Responsibility.

Specific responsibility for certain aspects of the quality management system is assigned to "top management". Often, of course, the work will be delegated - but top management remains accountable; they cannot just point the finger if things go wrong.

This includes responsibility for determining and meeting requirements, for customer satisfaction, and for ensuring internal communication regarding the effectiveness of the Quality Management System.

See ISO 9001:2000 sections 5, 5.2, 5.5.3

A.6. Quality Management

The Quality Manager ("Management Representative" in ISO 9000-speak) now has to promote awareness of customer requirements throughout the organisation.

The periodic Management Reviews need to consider customer feedback, process performance and product conformity, corrective and preventive actions, and changes that could affect the Quality Management System.

See ISO 9001:2000 sections 5.5.2 c), 5.6.2

A.7. Quality Policy and Objectives

Under ISO 9000:1994, the Quality Policy was often treated as a platitude with little application to the company’s actual activities. In the new standard, the requirements have been expanded and sharpened. You will need to review your existing Quality Policy, to make sure it complies.

Quality Objectives were referred to in the old standard, but little attention was paid to them. ISO 9000:2000 places a new emphasis on both overall company Quality Objectives and individual project Quality Objectives.

See ISO 9001:2000 sections 5.3 and 5.4.1

A.8. Configuration Management

At first sight, this may appear to have vanished from the standard. Not so!

It is covered by the requirement for "identification and traceability", which is now - confusingly - included under Production and Service Operations.

See ISO 9001:2000 section 7.5.3.

A.9. Other Changes

You must take immediate action to prevent defects when the need becomes apparent, not keep such action back to the next Management Review (ISO 9001:2000 sections 8.5.3 c) & d)).

You must re-evaluate suppliers from time to time, not rely on a once-in-a-lifetime initial evaluation (ISO 9001:2000 section 7.4.1).

A.10. Further Information

The above is a brief general summary of the main requirements of the new standards that will affect most software organisations. For specific advice as to the effect of the changes on your organisation, please contact the author.

Appendix B. TickIT versus ISO 9000

B.1. The Reasons for TickIT

The TickIT scheme makes certification of software companies' Quality Management Systems more credible than does ISO 9000 alone. But TickIT certification is still, nominally, to the ISO 9000 standard.

Before TickIT, software certification to ISO 9000 or BS 5750 gave little assurance as to the quality of the supplier's products or services. This was mainly due to a lack of uniformity in the interpretation of ISO 9000 for software, a lack of qualified auditors with a good understanding of software, and deficiencies in the certification process.

TickIT aims to remedy these shortcomings. Originally set up by the UK Department of Trade and Industry, it has spread to many other countries (with the notable exception of the USA). In the UK it is now administered by the DISC TickIT office, part of the British Standards Institution.

B.2. What TickIT Consists Of

The main elements of TickIT are:

- Guidance. The *TickIT Guide* interprets ISO 9000 as it applies to software. In spite of recent improvements to ISO 9000, this is still necessary - have a look at ISO 9000 and you will see why!
- Assurance as to the competence of bodies authorised to grant TickIT certificates, and the auditors they employ. These are vetted, to ensure their competence in the software quality field, and registered.
- Confidence in the value of the certificate, through use of a defined assessment process, and regular surveillance and re-auditing of certified companies - on a timetable designed to keep pace with the fast-moving software industry.
- Use of the TickIT name and logo as a "seal of approval". These are legally restricted to certifications granted under the TickIT scheme.

B.3. When TickIT Applies

TickIT applies to almost all software-related activities, whether they involve product development, services or both. This includes software developed as part of some larger product, and in-house software developed purely for the company's own use. Full details are given in the *TickIT Guide.*

B.4. Is It Compulsory?

No. You can obtain an ISO 9000 certificate without going through the TickIT process. However, such a certificate may not carry the same credibility with prospective customers as a full TickIT certificate.

B.5. Further Information

The *TickIT Guide* and other materials are published by the DISC TickIT Office, 389 Chiswick High Road, LONDON W4 4AL, telephone 0208996 7427, fax 0208996 7429 or e-mail <`tickit@bsi-global.com`>. The TickIT web-site is at <`URL:http://www.tickit.org`>. (http://www.tickit.org/)

The author, as an independent Software Quality Consultant, will be pleased to provide advice on these matters and assistance in setting up a TickIT-compliant Quality Management System.

If you are thinking of using consultancy ...

I would welcome an opportunity to learn about your needs and suggest what I might be able to contribute. If you would like to meet, please contact me at the address given in the preface to this document.

Quality Management Bibliography

[1] Wayne A. Babich, *Software Configuration Management*: *Coordination for Team Productivity*. Addison-Wesley, 1986.

A very readable, and fairly short, book on the essential software engineering discipline of *Configuration Management.* Clearly explains the underlying principles, which you need to understand even if you are using Version Control software to track your configurations.

[2] Barry Boehm, *Software Engineering Economics*. Prentice-Hall, 1981.

A comprehensive reference book, packed with facts and figures - though now dated. The fault correction costs are quoted in *Software Inspection* by Tom Gilb and Dorothy Graham, Addison-Wesley, 1993, and elsewhere.

[3] Frederick P. Brooks, Jr, *The Mythical Man-month*, 20th anniversary edition. Addison-Wesley, 1995.

This 1975 classic is still the best description of the problems of large-scale software development. The current edition also contains Brooks’ 1986 essay *No Silver Bullet* and an assessment of more recent developments. Many of Brooks’ ideas have been adopted or adapted by more recent software development projects - see, for example, *[15]*.

[4] Mary Beth Chrissis, Mike Konrad, and Sandy Shrum, *CMMI*: *Guidelines for Process Integration and Product Improvement*. Addison-Wesley, 2003.

This is the definitive work on Capability Maturity Model Integration, which brings together the various strands of CMM that branched out of Humphrey’s original work *[10]*.

[5] Tony Collins and David Bicknell, *Crash*: *Learning from the World’s Worst Computer Disasters*. Simon & Schuster, 1998.

A good non-technical account, by two computer industry journalists, of several failures in major software projects. See also *[16]* for an update (2004); it would appear that not much has changed in the intervening six years.

[6] Philip B. Crosby, *Quality is Free*. McGraw-Hill, 1979.

The best-known work by the originator of the ’Zero Defects’ school of Quality Management, which has achieved wide popularity, especially in the USA - though it is less highly regarded in Japan *[13]*. A good source on Quality Costs.

[7] W. Edwards Deming, *Out of the Crisis*. Cambridge University Press, 1991.

A Quality Management classic. Deming was a founder of the post war Quality movement in both the USA and Japan. He severely criticises Western industrial practices, in a well-argued case that draws heavily on his own observations.

[8] DISC TickIT Office, *The TickIT Guide*: *Using ISO 9001:2000 for Software Quality Management System Construction, Certification and Continual Improvement*, issue 5.0. DISC (a division of the British Standards Institution), 2001.

This provides essential guidance on ISO 9001 for software organisations. It supersedes the TickIT Guide issue 4.0. See also *ISO 12207:1995: Information technology - Software life cycle processes*, which provides a taxonomy of software processes, activities and tasks, and *[12]*.

[9] John Guaspari, *I Know It When I See It*. AMACOM (a division of the American Management Association), 1991.

A short and readable introduction to Quality Management. In a light-hearted style, it succinctly conveys the main Quality principles.

[10] Watts S. Humphrey, *Managing the Software Process*. Addison-Wesley, 1989.

The original work on the Carnegie Mellon University Software Engineering Institute (SEI) Capability Maturity Model. The CMM is an approach to Software Process Improvement and audit against five specified ’capability maturity levels’. It is especially popular in the USA.

[11] International Standards Organisation, *ISO 9000:2000*: *Quality management systems - Requirements*. International Standards Organisation (also available as a British Standard from BSI), 2000.

ISO 9001, part of the ISO 9000:2000 series, is the definitive standard against which software organisations are audited. It includes substantial changes from the previous edition, *ISO 9001:1994*.

[12] International Standards Organisation, *ISO/IEC 90003:2004*: *Software engineering - Guidelines for the application of ISO 9001:2000 to computer software*. International Standards Organisation (also available as a British Standard from BSI), 2004.

This standard provides an interpretation, from a software organisation’s standpoint, of the quality management requirements of *[11]*. It supersedes *ISO 9000-3:1997*.

[13] Kaoru Ishikawa, *What is Total Quality Control? The Japanese Way*. Prentice-Hall, 1985.

Another Quality Management classic. Ishikawa was one of the Japanese pioneers, who built on Deming’s ideas to lead the world in Quality. His book presents an alternative approach to that of Western experts such as Crosby.

[14] J. M. Juran and Frank M. Gryna, *Quality Planning and Analysis*, third edition. McGraw-Hill, 1993.

A text-book by Juran, one of the leaders of the post-war Quality movement.

[15] Eric Steven Raymond, *The Cathedral and the Bazaar*, version 3.0, 2002. <`URL:http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar`> (http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/)

This paper contrasts the "bazaar" model of software development, adopted in Open Source environments such as that of Linux, with the more conventional "cathedral" model followed in the commercial software world. It is illustrated with a case history of a project, successfully carried out by Raymond using this method.

[16] Royal Academy of Engineering and British Computer Society, *The Challenges of Complex I.T. Projects*. The Royal Academy of Engineering, 2004.
<URL:http://www.raeng.org.uk/news/attach/215.pdf>
(http://www.raeng.org.uk/news/attach/215.pdf)

This is the report of a study aimed at improving the understanding of how complex I.T. projects differ from other engineering projects, with a view to identifying ways to augment the successful delivery of I.T. projects. It includes a number of case studies, and provides an interesting update on *[5]*.

[17] John Seddon, *The Case Against ISO 9000*, second edition. Oak Tree Press, 2000.

This work argues that ISO 9000 institutionalises an outdated ’command and control’ mentality and the discredited ’inspection-based’ approach to quality. Therefore, Seddon maintains, it fails to deliver improved quality, and has actually damaged many companies that have implemented it.

[18] Jennifer Stapleton, *Dynamic Systems Development Method*. Addison Wesley Longman Limited, 1998.

This book describes the Dynamic Systems Development Method, a popular version of the Rapid Application Development approach. Key features include incremental development of requirements throughout the project, incremental deliveries throughout the project, prioritising of requirements and "time boxing".

[19] Edward Yourdon, *Decline & Fall of the American Programmer*. Prentice-Hall, 1992.

This is a readable and provocative work on the problems facing the U.S. software industry at the time of publication. Yourdon has published a sequel, *Rise & Resurrection of the American Programmer*.

Index