



Introducing quality control after the horse has bolted

G. Saltmarsh

*CSC Computer Sciences Ltd, 4 Weybridge Business Park,
Addlestone, Surrey KT15 2HE, UK*

ABSTRACT

Quality personnel are often brought into Software Development Projects only when the lack of adequate controls has become apparent. Common causes of this are discussed, and basic steps that can be taken to regain control over a software development project are detailed. Two alternative approaches and problems that have been encountered with these are also considered.

INTRODUCTION

It is not an unusual experience as a Quality Analyst/Manager to be called in to a project that is perceived in some way to be out of control, and to be expected to work a piece of 'Quality' magic on it. The difficulty of this task is not usually appreciated by those who request the help, and results are usually expected by yesterday. In such a situation, what priorities should we set, what should we tackle first and what pitfalls should we seek to avoid? Each situation is different. There is usually a maze of politics to trap the unwary, and success is hard to achieve. Nevertheless, after being involved in a number of these situations and having experienced the results of differing approaches, I believe there are guidelines that can be given.

SYMPTOMS

The very fact that help is being called in some time after the project has begun means that there must be visible symptoms of a lack of control. As often as not there will be several such symptoms, probably drawn from the following list.

1. Confusion arising from people working from different versions of



44 Software Quality Management

documents, all thinking theirs is the current one.

2. Plans proving worthless because the work done on a product after it has been 'finished' appears to be greater than that needed to 'finish' it in the first place.
3. Specification documents being unusable because of internal or external inconsistencies.
4. Constantly moving goal posts creating eternal rework, e.g. the database structure on which a team is trying to build its product keeps changing.
5. Seriously missed milestones (or complete lack of milestones).
6. Teams working with different and conflicting assumptions, either because their efforts to have them clarified have been unsuccessful, or because they have never asked.

The problems have often come to light because some fresh mind has asked a 'simple' question, and finds no-one can answer it.

The underlying causes of these symptoms will be the failure of one or more functional processes, and it is these functions that need to be put right. Below are listed functions whose failure may be responsible for each symptom.

Symptom 1 - Configuration Control; Version Control; Library.

Symptom 2 - Planning; Quality Control; Reviewing; Sign Off.

Symptom 3 - Quality Control; Reviewing; Product Breakdown Structure; Dependencies.

Symptom 4 - Change Management; Planning; Business, System or Technical Architecture.

Symptom 5 - Planning; Tracking.

Symptom 6 - Issue Management.

It is most likely that the difficulties stem from failures in several of these functions. They are substantially interrelated, and a lack of awareness of the importance of any one is likely to be associated with a similar lack of awareness of the importance of the others. Alternatively it may be that it was not the lack of awareness of the importance of these

functions that was the root cause, so much as a lack of awareness that the required formality of such functions is size critical. What will be adequate for a small project may be quite unable to control a larger one. In this case also one would expect several of the functions to be at fault at once.

STRATEGIES

The solution on the face of it is simple. Put in place adequate procedures in all of these areas and we can quickly regain control. Oh, that it were so easy!

Every business will have its own standards and procedures, its own list of approved hardware and software packages, and its own culture. This will be true of the situation into which we have been brought. In some way the existing procedures will be known to have failed and thus be discredited, so we will be very lucky if we don't meet resistance to the introduction of new procedures. We all recognise the need to introduce a solid, comprehensive, consistent set of standards. We also know this cannot be done over one weekend. In fact we know that we are going to need several weeks or longer to get a comprehensive set of standards in place, even if we start with a set taken off the shelf. Prior to the introduction of a wide range of new standards, there will be demands for them to be reviewed, approved and authorised, and this can easily turn into a very long process. So do we go ahead nevertheless and make the introduction of these our top priority?

I suggest that before we manage to get approved and established such a comprehensive Quality Management system (which I am using as an umbrella description of the standards and procedures of these many and varied functions), we will have lost our credibility since the confusion we were brought in to fix will have continued to worsen. All we will be perceived to have done is to divert time from 'real tasks' to 'quality matters', and brought no visible benefit.

So do we delay the establishment of a sound set of standards and procedures and set out on an ad hoc basis to identify errors and correct them? Do we concentrate on removing the inconsistencies from the Requirements Definition, revalidating the Scope and Objectives and establishing traceability from Requirements to Logical Design to Physical Design, making the removal of errors from the products created so far the top priority.

At first sight this course also has much to recommend it, but unfortunately, as we clear existing errors, we will find that the same lack of control as brought problems before will allow fresh errors to reappear

46 Software Quality Management

elsewhere unchecked. The world does not stop while a software development takes place, and there is always the need for changes to be introduced. Without adequate control, this will quickly undo all the good work that has been done.

Somewhere between these two approaches a compromise has to be found. Of the two approaches described above, dare I say that it is the second which I believe is closer to the best way forward. I put it like that because it must surely be heresy in any Software Quality Management Specialist Group to accept anything less than a comprehensive set of standards. I don't believe it is heresy. The issue is not about what we aim for, but rather how we get there.

If things are known to be wrong we cannot afford to let this continue to worsen if our credibility is to be maintained. Rework is one of the ultimate demotivators. To be told that, as result of other people's errors your last five week's work must be scrapped and a fresh start made does not make one happy. I have experienced leading a team who were on their third set of system test plans, having over the previous six months produced and sent to the shredder the previous two sets. Saying that they must have become quite practised at the task by then, was not the way to make friends and influence people! The circumstances that had given rise to this were similar to those we are considering. When the project began to run into problems around the Functional Specification stage, a 'Quality Group' was formed. The approach taken by this group was influenced by the politics of the situation and this resulted in much time being spent generating and introducing new control procedures. By the time these were introduced disillusion was widespread.

It is vital to phase in controls so that, from the first moment, there is clear and effective action which can be seen to be bringing order out of chaos.

THE PHASED APPROACH

I have divided this into three phases - Immediate First Aid, Securing the Recovery Environment, and Providing the Treatment.

Immediate First Aid

The first step must be to check any further loss of control. This requires two things - a Controlled Library and a review and acceptance process.

Controlled Library This must be able to hold any product which may be created during the project, and be able to hold it in such way that it cannot be altered without the knowledge and agreement of the Librarian. Although once this library is established, a fairly urgent task is to capture

copies of all the key products already in existence, first and foremost the library is being created to hold the definitive version of everything produced from now on.

Review and Acceptance Process This second arm of the immediate first aid is the insistence that no product is to be considered complete until it is delivered to the Librarian accompanied by the record of its final review. This review will have consisted of a meeting of appropriate reviewers to check out the product. The product will have been made available to the reviewers in advance, and they will have come to the meeting armed with their comments. The meeting will consider the comments and decide how each is to be dealt with. All actions required to bring the product to an acceptable standard will be agreed and recorded. If any of the reviewers wishes to register disagreement with the majority decision, this will be recorded also. At the end of the meeting the reviewers will make a final decision on the future progress of the product from three possibilities:

- 1) Accept product as is.
- 2) Accept product after actions have been completed.
- 3) Re-review required.

The most usual decision will be the second, where the product is to be accepted once the remedial actions specified in the record of the meeting have been carried out. Completion of all the actions will be checked by the person nominated for this purpose by the meeting, who will then sign to this effect on the review form. Some actions may need checking by a particular person or group, in which case the action point on the meeting record should state this, and until this checking has been done the form cannot be signed off.

The format of the review form is kept simple, and Appendix A1 provides a model. The form is part completed before the meeting and part at the end. The Action Points of the meeting are recorded during the meeting on the blank reverse side of the form and on plain paper continuation sheets (numbered sequentially) as necessary. An essential part of the process is for each reviewer present at the end of the meeting to sign the review form to show their agreement to the record of the meeting, both action points and decision. Thus each action point must be recorded at the time it is decided upon. This requirement for each reviewer to sign their acceptance of the meeting's decisions heightens the level of commitment and is an essential part of the process.

With this review and acceptance of products, linked to a controlled Library, we now can identify completed tasks. From this point on work

48 Software Quality Management

will be significantly more controlled, and even if rework of it is required, at least it will be known exactly what it is that is to be reworked. The controlled library will hold the definitive version, and the review record will give details of those involved in the final stages of its production.

It is perhaps worth adding here that a simple hand-written paper system of recording the review is very powerful and can be instituted with minimum delay. All that is needed is a stock of the forms, a ring binder and some 'log' sheets (see Appendix A2). Prior to a meeting, a new entry is made in the log and the corresponding sequential log number assigned to the review. Following the meeting, copies of the meeting record are made and distributed as required, and the original form stored in the ring binder. The log sheet allows easy audit of those reviews which are still incomplete, since the completion date is only filled in when final sign off has been achieved. In the case of 'Accept as is' and 'Re-review required' that will be on completion of the meeting, but in the case of 'Accept product after actions have been completed', final sign off is not given until the actions from the review meeting have been completed. Since a Librarian will be needed to maintain the Controlled Library, the same person can also look after the Review Log and Review Records.

Together the Controlled Library and the Review and Acceptance Process bring immediate benefit.

Securing the Recovery Environment

The next step is to define and make effective two more processes - Issues and Change Management, and at the same time to set up a Project Library.

Consider first the two processes, Issues and Change Management. Registration of either an issue or a request for change will start with the completion of a form. The forms will log either the issue which needs resolving or the change that is desired, together with details of subsequent processing until either the issue is resolved or the change made or rejected. The forms can be handled electronically if a suitable networked system exists, otherwise a paper-based system does the job quite adequately. In either case use of a spreadsheet as a register and status recorder is to be recommended.

Each Issue and each Request for Change has a life of uncertain length at the end of which it will be closed off. The management process in both cases is about reaching this closing off point as quickly and with as little expenditure of effort as possible. It is about directing and monitoring progress. Hence the need for a spreadsheet or similar device to allow the Issues or Change Manager to know who is dealing with each item and what the current state of its progress is.

Change Management This process is a natural result of having a Controlled Library. Items are in the Controlled Library because they are used as a basis for later stages of work. A specification forms the basis for design, a common routine is a building block in number of software modules, a completed software module is the input to series of tests. Change to any of these will ripple into other products. Stability is needed, and also the possibility of change must be allowed for. Essentially Change Control must achieve two things. It must allow the full impact of a change to be known (as far as possible) before the decision is taken to allow the change, and it must ensure that all those whose work is based in any way on the product to be changed are kept informed both of the proposal to make the change and of the availability of the changed product when it has been completed.

Any form of Change Management is better than none. So a first step is to have a simple Request for Change form on which the proposed change is specified. The form must also allow for identification of any other products affected, authorisation (by the Project Manager or delegate) for the change to be made, and documentation of exactly what change is made, how the new product is identified from the old, and how the completion of the change is publicised. The form shown in Appendix A3 will support this. Its structure is very basic and, when time allows, this should be enhanced to include impact assessment sections and further status recording. However on initial introduction that does not need to be formalised, and it is part of the approval authority's job to ensure that adequate impact assessment has been carried out before the change is authorised.

Issue Management Issues systems can suffer from too much complexity. What is needed is a formal method of getting timely resolutions of concerns, and moreover resolutions that will stick. The bare essentials are a form to be used to raise the issue, and a Manager to pick the people who can provide the solution and to hassle them until they do.

Issues should be allowed to range from the trivial to the critical. What is important is that the time given to the trivial should be minimal, while the time devoted to major issues may need to be considerable. It is better to accept trivial issues and resolve them at once (since the fact that the issue is raised at all means that it is not trivial to someone) than to try and prevent them being raised. The avoidance procedure usually takes longer than resolution.

An example of a trivial issue in an over complex issue process concerned the size of the typeface for documents used within a programme of projects being carried out at many sites which were widely dispersed but electronically networked together. There was no firmly

50 Software Quality Management

established standard and some people used 10 pitch and some 12. The tendency around the project office was to use 12 pitch, but some locations could only print in 10 pitch. It was a trivial issue, but months after being raised it still remained unresolved.

I believe it is worth allowing two types of issue, those whose scope goes outside the jurisdiction of the person raising it (e.g. it may affect two teams and thus require either agreement of both team leaders or decision from a higher level to resolve it), and those which require the passage of time before they can be resolved (e.g. "When the detailed design stage is reached it must be remembered that"). The former is the more usual type, but since regular monitoring is an essential part of an effective Issues system, lodging within the system things to be remembered at a later date allows them to be recalled at the right time, and the regular monitoring ensures that this happens.

A basic Issues Form is shown in Appendix A4. Recording and logging the issue is the easy bit. Ensuring that issues get resolved is more tricky. It is fatal to set up a system for the collection of issues unless there is also a determination that those issues that are collected will be resolved. However do not allow this to put you off, as resolving issues is one of the principal purposes of management.

Project Library All libraries are difficult to run. There is a permanent conflict between easy access and loss of control. This is equally true of Project Libraries, and a middle course has to be trodden. The problem is lessened by the existence of the Controlled Library in which items which must not be lost can be stored in safety. As well as the Controlled Library a project does need a more open library where anything it might need later can be lodged. This library does need to be structured. A good starting point for this is the first level of subdivision proposed by PRINCE - Management, Quality and Technical. A second level of subdivision is suggested in Appendix B, but this would need tailoring to particular needs.

Providing the Treatment

By now readers should be wondering when planning is going to get a mention. Planning work and then tracking it is an essential for control. Planning and tracking the project should have been going on from the outset, but this may not have been the case. All too often plans and reality move so far apart that any sort of tracking becomes impossible. This may well have been a major cause of the project's problems in the first place. If you don't know what should be being produced and when, it is hard to know, when you do get a product, whether it is what it should be or not!

In spite of the central place of planning, I am convinced that, in the

circumstances of an out of control project, a higher priority must be given initially to the provision of at least rudimentary forms of the five elements that have been mentioned above (Controlled Library, Review and Acceptance Process, Change Management, Issues Management and Project Library) than that given to getting the planning and tracking back on course.

The reason for giving planning this lower priority is that the time required to establish the five essential elements is quite short, and if these are implemented everyone involved is immediately aware that things have changed. On the other hand, to get a firm grip on planning and to straighten that out is likely to take several orders of magnitude longer.

Ideally it will be possible to start bringing the planning under control at the same time as these other activities, but if it does come to conflict of resources, then planning must take second place.

The first priority in restoring the planning must relate to the current work - what is being produced, when and by whom. Then must begin the long task of going back to the start and tracing through from the initial objectives and scope (which may themselves need revalidating), and validating all that has been produced. As each product is validated the valid version must be taken into the Controlled Library, any changes having been made under the Change Control system. In parallel with the validation of earlier work must go the identification of the complete set of controlled products (perhaps in the form of a Product Breakdown Structure), and also the identification of the Hardware and Software configurations and the establishment of Configuration Control.. All this must be planned systematically. There is not room here to detail how this planning should be done, and this is in any case a complete subject in itself.

IN CONCLUSION

What has been described above represents a task force's first steps. Consolidation will be needed, but the early successes gained by this phased approach makes the job of growing the system of controls much easier, as people will be experiencing the benefits of the stability that adequate control brings. Progressive improvement and enhancement to the five rudimentary controls established at the outset can be made.

It will emerge that some development work needs to be put on ice to free up the resource to validate earlier work, and some development work will need to be redone, but experience has convinced me that this phased approach is effective. Putting in 'cheap and cheerful' first aid control to allow breathing space, and only then turning to past error correction,

52 Software Quality Management

proper planning and the progressive enhancement of the system of controls is the best way forward. It is significantly more effective than either the approach that says "We must fix past errors as top priority so we can keep the momentum going. Then we can get the control infrastructure in place." or the approach which says "Get a complete infrastructure together with all its standards and procedures first. Then by implementing that, we can bring everything back on track."

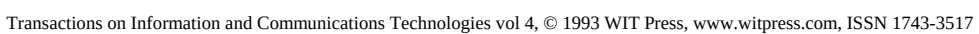
In the bolting horse analogy, first close the boundary fences to prevent the horse going further afield, then fit some basic means of securing the stable door, and only then spend time fetching the horse back. At this point enhancements to the mechanisms for preventing the horse bolting again can be made.



APPENDIX A FORMS

A1 Review and Acceptance form

REVIEW & ACCEPTANCE	
Review Id: _____	
Product Name: _____	
Product Ref: _____	Version: _____
Date: _____	Start Time: _____ End Time: _____
Reviewers:	
<u>Name</u>	<u>Signature</u>
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
Review Decision(tick one box)	
☐	Accept product as is.
☐	Accept product after actions have been completed.
Completion sign-off by: _____	
Signature: _____ Date: _____	
☐	Re-review required.

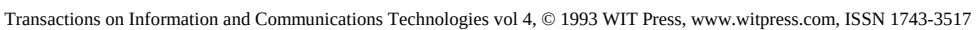


A2 Review and Acceptance Log form

[illegible]

A3 Request for Change form

REQUEST FOR CHANGE	
R.F.C Ref: _____	
Requested by: _____	Date: _____
Description of Change:	
Reason:	
Date by which change is required: _____	
Controlled Items Affected (incl. version nos.):	
Authorised by: _____ Date: _____	
Implemented by: _____ Date: _____	
Details of Implementation (incl. version nos. of changed products):	
Completion notified by: _____	



A4 Issue form

ISSUE	
Issue Ref: _____	
Raised by: _____	Date: _____
Issue Description:	
Date by which resolution is required: _____	
Working Assumption:	
Details of Resolution:	
Authorised by: _____	Date: _____



APPENDIX B

Library Structure

A second level of subdivision might be:

Management

- Project Background
- Project Initiation
- Budget
- Plans
- Staffing
- Progress Reports
- Meeting Minutes
- Correspondence
- Telephone Records
- Standard Forms

Quality

- Standards and Procedures
- Quality Criteria of Products
- Review and Acceptance
- Audits
- Requests for change
- Issues
- Off Spec Reports

Technical

- Technical Manuals