Transactions on Information and Communications Technologies vol 8, © 1994 WIT Press, www.witpress.com, ISSN 1743-3517

Empowerment of software professionals for quality

A. Jones

School of Computing and Mathematics, University of Teesside, Cleveland, TS1 3BA, UK

ABSTRACT

Empowerment is essential for proactive, quality-inspired individuals who can implement QMS directives in a capable and cost-effective manner.

This paper will present a particular philosophy, curriculum and teaching style that together inculcate and elucidate this human factor of self-realising empowerment. The course, an advanced MSc in Software Engineering, has evolved under various funding arrangements of the Department of Employment, and had to meet quite severe performance targets on an annual basis. With a curriculum aim that includes specialised bottom-up technical processes as well as general top-down quality management systems, this set a challenging task for the course designers.

The standard MSc curriculum steps of taught units and an individual project have been enhanced and expanded with notions of learning contracts, quality plans for short projects, tutor-driven audits, student-driven review and inspection techniques, peer and self-appraisal, and student audit of a set of undergraduate software engineering projects.

The effectiveness of the programme is assessed with reviews and comments from the class, former graduates, the industrial sponsors who provide many projects and a survey of all graduates since course inception in 1990.

Introduction

Software engineers have responsibilities for public safety. Education courses traditionally address such responsibility by seeking to provide a solid grounding in principles, a disciplined training in thoroughness and completeness and an awareness of professional responsibilities. (Parnas [12])

The advanced MSc course at Teesside University has been running for four years now. Our design parameters were very similar to Parnas's: to present what principles we could, to emphasize thoroughness and completeness as well as intellectual argument and to provide software engineers who are attuned to manoeuvring and defining the concept of quality as a natural objective of their careers. Work by Ford and Gibbs [6] and Basili and Musa [1] also helped form our programme.

This paper will concentrate on our efforts to bridge the gap between theory and practice for software quality. In particular, we have sought to remove the stigma of bureaucracy that still bedevils quality management systems.

Course design

Quality is not a single idea but a multidimensional concept. Much has been written of quality models, quality metrics, quality certification and good practice for quality conscious professionals (Deeming [4], Fenton[5], Gilb[8], Pressman[13], Pfleeger[14] and TickIT[15]). A course that presents any particular slant, say ISO9000-3 and TickIT, will at best raise more questions than it answers and at worst be swallowed whole by students without any apparent effect save that of regurgitation at examination time followed by perpetual amnesia or aversion.

Our goal was to produce critically aware graduates and our mechanism was and is to give authority and power to students and trust in their self-interest to create a learning "pull" rather than a centrally-controlled teaching "push". These ideas are not new. Knowles [11] has described his work with adult learners: the key is negotiated agreement between learner and teacher about what is to be learned. This is the learning contract. Negotiation is the focussing of motivation in the learner; this is the empowerment process.

Basili and Musa [1] comment on the breadth of descriptive models needed for software engineering education, and in particular draw on work of Curtis, Krasner and Iscoe [2] to justify their emphasis on application domain knowledge as a principal factor to distinguish different capabilities of software engineer.

The design of this course addresses these issues, as well as focussing on the requirements of our customer, the Department of Employment operating through its various programmes of training.

Philosophy of the Course

Software is a means to an end. It serves the needs of its users - business, commercial and industrial. There are no 'right' answers; there are, instead, a set of possible answers whose suitability and application must be judged by

reference to the user's requirements. This judgement of degree may appear to apply equally to quality, but this is too simple and misses the essential orthogonality of quality. Users needs are satisfied by application of controlled quality, and this is quite distinct from the visibility of a certificated Quality Management System.

Curriculum

The curriculum is constructed as a vehicle for delivering application domain knowledge within a form a negotiated learning to effect empowerment. Reasons of cost and time, as well as our own inexperience, lead to the development of a hybrid negotiation process, where the learner is gradually given increasing authority to set the goals of the learning. Table 1 gives the technical areas addressed in the curriculum.

Table 1 Curriculum

stage	weeks	Units of study MSc
Stage A	4	Project management, Formal methods, Structured Analysis, Advanced Programming
Stage B	8	Design methods for Information Systems, Quality Assurance and Project Management, Object-Oriented Techniques
Stage C	10	Design methods for Concurrent Systems, Audit and Conformance to QA standards, Software Engineering Support Environments
Stage D	18	Individual Project leading to MSc thesis

Teaching Style

Short group projects

Figure 1 shows the basic model for software engineering group projects adopted within the Software Engineering division at the University of Teesside and operated as a learning contract. Tutor-specified and assessed goals of product and process are added to the team-specified and assessed goals of group contribution.

The design of this rubric owed much to studies of adult learners (Knowles [11]) and suggestions on alternative assessment regimes (Gibbs [7]).

Group organization and roles are decided by the group in the light of classwork on communication and management, and influenced by the goals to be attained. Each team has a staff tutor as supervisor and consultant; and where possible another staff tutor as client for arbitration of product specification, acceptability and satisfaction.

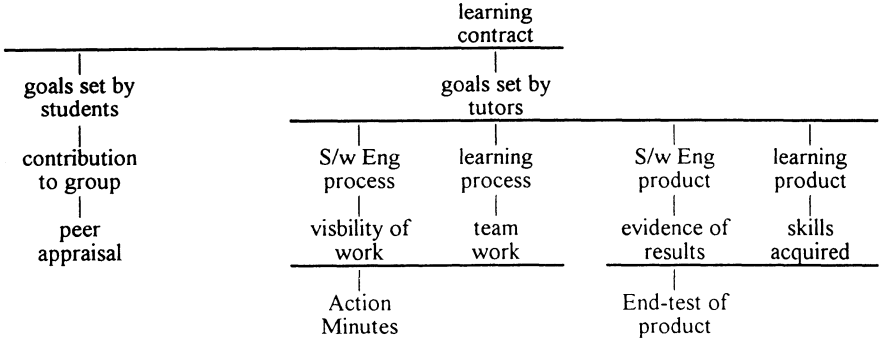

Figure 1 **Rubric for Group Projects**

Each group project is constructed to give practice and in-depth experience in an application domain (technical process) and in quality management (teamwork, planning , standards and verification). Table 2 shows the progression of application areas, per group project.

Table 2 **Primary vocational skills - Group Projects**

Group project	Application domain/skill
1	structured methods, team work
2	Formal methods, methods integration
3	Methods integration, concurrent systems

Groups meet to take decisions and allocate work; records of such meeting, the Action Minutes, form a central part of the evidence for assessment of process. We have operated this scheme for several years now (See Jones and Birtle [10]).

Under the original view of this contract, goals of cooperation and contribution are set by the team themselves. Leaving the choice of peer appraisal criteria to the teams has resulted in much resistance and adverse comment from students because of imprecise targets such as *leadership* and *commitment*. These goals are now given for projects 1 and 2 and become the basis for peer assessment which we intended primarily to motivate and encourage but also to discriminate between individuals in a group.

To promote the experience of quality and product management, there is a manager appointed per team, with a rota to ensure all members of the class experienced this role. We operate a complicating factor whereby we reform the membership of each team after one week and install a new manager, ensuring

that 'old' managers do not stay with their original team. This is to mimic the 'real world' by bringing new people on to a project. This is similar to devices described by Dawson [3].

Student-driven review/inspection
This is the mechanism for verification and also provides both agenda and forum for assertive action, another building block of empowerment.

Personal learning goals and self-appraisal
The assessment incorporates the setting of *personal learning goals*, and the creation of a peer appraisal system based on degree of achievement of these goals. We are very positive about the delegated power achieved by this mechanism. Individuals recast their targets for assessment in areas which students have traditionally avoided in group work, namely the high-risk areas. We have also devised an assessment regime to reward the extent of achievement of personal goals; in other words, to reward risk-taking.

Tutor-driven audit
The function of tutor-driven audit is to place quality attitudes in the context of technical processes. Brief quality criteria are set for each group project. These typically involve definition of verification working pracices, and documentation standards for design notation, and are in addition to the requirement for a project plan. The empowerment goal is to prioritize the review/inspection process and to crystallize the role of design notation as a documentation standard. The need to evince control of verification and to justify the use of a design notation puts these embedded-quality control activities into a simple perspective. Tutor audit closes the Plan-Do-Check-Act visible-quality loop (Huda and Preston [9]) by requiring corrections on issues raised at audit. Realistically, documentation audit issues can be closed by a subsequent corrective action; but in our short group projects process audit is fedback without the opportunity or obligation for correction. In this way, a cornerstone of embedded quality is devolved to the student by example.

Completing the process of empowerment

By the mid-point of the year-long course, students have experienced negotiating and delivering on learning contracts and personal goal plans, presenting and discussing technical topics in student-led seminars, working under their own management and tutor-audit in short group projects with close quality and management control and extensive class work and assignments on technical domain topics.

Three major steps remain to complete the process of empowerment:
a) responsibility to audit and comment on the work of another team
b) responsibility to set their own quality criteria and conformance mechanisms
c) the journey to fulfilled empowerment during the individual project

Student audit The term "proctoring" is used to mean the supervision of junior students by senior students. Proctoring has been incorporated as a first step in dealing with real customers, an essential experince without tutor support if authority and power are to be successfully devolved.

Ground rules for the conduct of an audit by teams of MSc students of a two-semester case study running in the final year of an undergraduate BSc programme. The customer commissioning the audit is the Case Study Tutor, to whom reports are made about adequacy of documentation for each case study team. The audit runs over a 10-day period with pre-reading, process inspection, closing meeting and final report. The undergraduate teams are offered right of reply in an open meeting to comment on the conduct and comments of their auditors. In the history of proctoring on this course, no team has declined the opportunity to reply. Finally, in a closed session, the MSc students are debriefed on their process and awarenesses arising from this exercise. The most frequent comment concerns the realization of responsible power - that audit helps things to work better. This is to be compared this with the regrettable but too common pre-audit feeling expressed in the comment "Now we'll get our own back."

Self appraisal Self-appraisal contributes increasingly towards assessment, so that by project 3 students are responsible for a significant part of their assessment, but we expect a full report on the criteria and methods used. The supervising tutors corroborate and moderate these self-appraisals.

Individual project Inseparable from the course philosophy is the view that every project should have a client. We have been very successful at procuring live project opportunities (see Tables 3 and 4).

Table 3 Provision of projects

session	students recruited	internal projects	external projects	total
90-91	16	9	24	35
91-92	18	14	29	43
92-93	20	22	39	61

Care is taken to vet each project for capability to deliver MSc-level reports. Clients are invited to nominate a problem that neatly captures a business IT concern, a wish-list item or a technology transfer "what-if" issue. The problem is then prefaced by a requirement on the student to research with background reading, and postscripted with a requirement to make a critical evaluation of results and method. Each approved project is capable of a full lifecycle of hypothesis, design/build and evaluation. The client gets the product of the build phase. The student gets an MSc thesis.

Frequently the client's quality management system is woven into the fabric of the project, and the student has had to defend and explain all aspects of the project to an internal QA team as well as to a final *viva voce* examination panel.

Table 4 Projects selected by students

session	students at Stage D	industry client	on-campus client
90-91	12	8	4
91-92	16	11	5
92-93	20	18	2

Our guidance to academic supervisors describes the supervision process as one that moves from giving confidence at the start of the project to sharing confidence at tbe the time it finishes. Finally, the students themselves put on an Exhibition of Projects and present their work to a wide audience of examiners, other students and the Steering Group.

The course and especially the individual project are strongly supported by an industrial Steering Group, currently comprising representatives from British Telecom, SERCo Ltd, IBM, British Gas, and National Power plc. Senior managers give much time and effort to the maintenance of liaison and we much appreciate this.

Effectiveness

Each year we report back to our funding source on the outputs we achieve with each year's cohort. Despite operating in very straightened times economically, we have been able to report a consistent 80% success rate at students located in relevant employment or further education within three months of leaving the course. Recruitment has not been wholly from undergraduate degree-holders. About half the recruitment has been from mature students with non-standard qualifications. Their successes have been our successes also.

Each year we record the Exhibition event in the form of a brochure or videotape. Some of the comments are:
"..project (is the) most enjoyable part of the course." (a student)
"..technical content very high.." (a member of steering group)
"..project achievement in 16 weeks excellent..relevant to industry.." (another member of steering group)
"..students well motivated ..clear foundation." (a client)
"..teamwork is at first difficult at the same time as a project.." (a student)
"..looking for recruits who can hit the ground running. This course provides them." (senior manager, member of steering group)
"..Teesside is one of the few places offering this type of course." (a mature student)

Our Steering Group has remained a source of ideas and strength. Their continued support for the course also speaks for its effectiveness in their eyes.

References

1. Basili, V.R. & Musa, J.D., The Future Engineering of Software: A Management Perspective, *IEEE Computer* **24(9),** 90-96, (Sept 1991).

2. Curtis, B., Krasner, H. & Iscoe, N., A Field Study of the Software Design Process for Large Systems, *Comm. ACM*, **31(11)**, 1268-1287, Nov 1988.

3. Dawson, R.J., Newsham, R.W. & Kerridge, R.S. (1992) Introducing new software engineering graduates to the 'real world' at the GPT Company, *Softw Eng Jour* 7(3), 171-176, (May 92) .

4. Deming, W.E., *Out of the Crisis*, MIT Center for Advanced Engineering Study, MIT Press, Cambridge, Mass, 1986.

5. Fenton, N.E., *Software Metrics - A Rigorous Approach,* Chapman and Hall, 1991.

6. Ford, G.A & Gibbs, N.E., A Master of Software Engineering Curriculum, *IEEE Computer* **22(9),** 59-71, (Sept 1989).

7. Gibbs, G. (1986) *53 interesting ways to assess your students* Oxford 1986.

8. Gilb, T., *Principles of Software Engineering Management*, Addison Wesley, New York, 1988.

9. Huda, F. & Preston, D., KAIZEN: the application of Japanese techniques to IT, *Software Quality Journal* 1(1), 9-26, (March 1992).

10. Jones, A. & Birtle, M. (1989) An Individual Assessment Technique for Group Projects in Software Engineering, *Softw Eng Jour* **4(4),** 226-232, (July 1989).

11. Knowles, M.(1990) *The Adult learner: a neglected species* 4e, Osborne, 1990.

12. Parnas, D.L., Education for Computing Professionals, *IEEE Computer* **23(1)**, 17-22, (Jan 1990).

13. Pressman, R.S., *Software Engineering, A Practitioner's Approach*, McGraw Hill, New York, 3e, 1992.

14. Pfleeger, S.L., *Software Engineering, the Production of Quality Software*, Macmillan, New York, 1991.

15. TickIT Guide, DISC TickIT Office, 2 Park St, London W1A 2BS.