

Part Two: Background

We have come to believe that activities labelled ‘*Transfer of good practice*’ and ‘*Dissemination of teaching and learning*’ are both more subtle and more complex than the common rhetoric suggests. Our understanding has grown from our twin viewpoints, acting both as disseminators (facilitating the transfer of materials/practices) and as practitioners (wanting better solutions to persistent problems) and the interaction between these roles.



Transfer expects transformation

Evaluating success in transfers is not easy:

“... whether a transfer is good or bad is contingent: it has to be judged on the merits of a particular situation. Ideally, one would perform a full cost-benefit analysis on every case of transfer or transfer failure to say if it was appropriate. In practice, this can be very problematic because costs and benefits are delayed, hidden and not quantifiable with confidence”[1]

In particular, in our evaluations, we encountered something unexpected: nothing emerged the same as it went in. No practice was “transferred” in the sense that the importer institution replicated the practice as it was undertaken in the exporter institution. There were recurrent phrases in the evaluations concerned with the changes that were made. “[they] had to implement a version of the bundle themselves”, “Any materials used would have to be modified for local conditions so I was not looking for directly transferable materials” and, “As the package provided was sketchy there was plenty of scope for putting our own stamp and interpretation on it”. [2]

At first we regarded this as “transfer failure” but, as it occurred so often, we started to think of it as a characteristic, and came to the view that if a practice had not been adapted or otherwise changed in the process of adoption then “transfer” had probably not taken place. We came to believe that the question “How have you changed this?” was a metric for transfer success. If the question could not be answered, then there was no transformation: without transformation there was no transfer. Our view changed so that for us “transfer” was not essentially concerned with the exchange of ideas and materials but with transfer of the *ownership* of those ideas and materials. Although surprising to us, these behaviours have been previously documented:

“Diffusion of skill and knowledge from one community to its neighbours and neighbours’ neighbours constitutes the central process of human history. Ever since significant differences in skills arose among separate human groups, borrowing back and forth has taken place whenever someone saw a real or apparent advantage in doing so. Borrowing nearly always involved modifying what was borrowed to make it fit smoothly into a different set of skills and customs”

and

“In real situations, borrowing provokes invention, when the new does not quite fit what was on the spot already; and invention provokes borrowing, whenever what has been invented proves attractive or threatening to others”[3]

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



In the transfers we observed we found that importers were keen to choose only that which was going to advantage their own situation and were careful and cautious in their choices. Secondly, we observed the *borrowing provokes invention* activity in almost every exchange: “it will be used in the context of individual third year rather than second year group projects”, “If the idea is good it will be adopted and adapted to cope with local constraints” and “The benefits of teaching practices are always oversold. The bundle allowed me to focus on the problem and to gain some improvements but has not provided the total solution it offered. But I did not believe it would to start with (no silver bullets)” [2]

This act of “tailoring” a piece of practice (which had *already* been abstracted for transfer) seemed to be integral to the process of transfer, and created part of its value. There are *borrowing provokes invention* practices evident in Part Two. The attentive reader will note that some bundles, packaged for different aims and appearing in different sections, are actually different aspects of the same piece of practice. We have kept these in because we recognise the value of the process that they represent. There are two implications from this formulation of transfer. The first is that you should expect to change anything that you import. The second is that, as an exporter, you have to “let go” and abrogate your ownership of the practice. This is not necessarily easy, especially in an academic environment, where reputation rests on claiming ownership of ideas and practices, but it seems to be the way effective transfer works.

Focussing on solutions

In undertaking transfers you have to be aware of the pressures on practitioners and the limited enthusiasm they may have for changing their practice. This has been succinctly described with regard to medical doctors and their adoption (or not) of evidence-based medicine, where

“the emphasis on the need for evidence in medicine, and better transmission of information, needs to be balanced by a recognition that most general practitioners are *pragmatic, averse to innovation, and already feel overwhelmed with information.*” (emphasis added) .[4]

Educational practitioners, too, are resistant to prescription and may be “averse to innovate” in the face of explicit expectations that they import “best practices” in teaching and learning from elsewhere. So you have to find a form, a “packaging”, for transfer materials which does not prescribe or patronise and equally does not “overwhelm with information”.

We use a form which emphasises what is good and advantageous, which addresses real needs and which provides sufficient detail without being prescriptive. We were influenced in our choice of “packaging” by the work on patterns and pattern languages[5, 6] and devised a pattern-like form which is solution-focussed, allowing you to assess the benefits of the practice and decide whether it would be useful for you to adopt it. This form also allows us to describe a real implementation of the ideas

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

and materials without second-guessing whether they would be appropriate in your context¹.

Dealing with context

To identify a need for change is to identify only half the problem. Practitioners do not exist in isolation nor work in identical circumstances. For transfer to occur, not only must you want to change your practice but you must also be able to do so. Your context has to permit you to change.

We found context to be a very big problem indeed. Practices for transfer are not recipes, not some set of instructions which you can follow to obtain a guaranteed result, precisely because of context. Context is a difficult thing to grasp, to pin down, but can wreck many otherwise well-intentioned efforts at transfer. In the same way that it is unhelpful to say that *Wine should be as cheap in England as it is in France* without accompanying the statement with a substantial investigation into the reasons—reasons of history, geography, taxation regimes and culture—it is just as unhelpful to say *You should do projects in this way* without considering all the details of context that they rest on.

And yet, how can that context be adequately described? It is comparatively easy to say “this came from a prestigious research-oriented department” or “this came from a teaching university with large class sizes” but that sort of information is insufficient (and at the wrong scale) when what you want to change is how you do projects. What does it depend on? Smart kids? Small classes? Or something which the original teachers didn’t even consider: “We do it this way because years ago we tried it another way and it didn’t work” or “This fits because it builds on something I know my colleague does in a previous class – but only because I used to teach on that course”. For effective transfer it is necessary that your chosen pieces of practice are not only fit for purpose, but fit to culture.

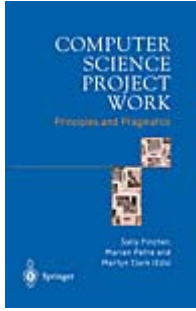
Initially, we tried to list everything, at every level, which impacted on the practice we were describing. However, the unpicking and uncovering of these dependencies proved to be, in practice, not particularly valuable. We were not the first to discover the twin horns of this particular dilemma:

“... we tried to give a point-by-point analysis of all these ‘real-world’ problems, in an effort to show, one by one, how they could be solved ... But somehow, no matter how we wrote [it], it always seemed thin. Either the solutions we proposed were too concrete and specific, or, in other versions, too vague and general. Somehow, however we wrote this chapter, it never seemed entirely convincing, even though it actually gave detailed answers to all the the specific points which might arise” [7]

¹ The form is “pattern-like” but bundles are not patterns. A pattern specifically describes an invariant quality which has been abstracted from many different examples. Our form inverts this to describe a specific piece of practice, which is inevitably transient.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>





Transfer Checklist

Part two, therefore, has been constructed to maximise the chances of transfer of practice, taking into account the both the “disseminator-push” and the “practitioner-pull” against three rules of thumb:

- Look to transfer small pieces of practice. Anything too large will infringe local context and be broken down into smaller things anyway.
- Focus on the solution, not the problem
- Describe the practice as you do it—not in the way you think someone else should.

Part Two: Introduction

We have divided Part Two in six thematic sections. The structure of part two reflects

- what we think project work involves
- why we think these are the important aspects

For example, we have no section on “deliverables”. We believe that, if you get these six aspects right, then you get deliverables “for free”. That is to say that technical objectives are easily achieved if the projectwork experience is correctly (and appropriately) structured. The sections are:

Allocation

When undertaking projects, unlike lecture courses, there is usually an element of choice (for both staff and students) which must be managed by a departmental process. How that choice is exercised, and how allocations are made, can drastically affect the outcomes of projects.

Supervision

There are many ways to act as a supervisor for a project. However, in the vast majority of cases, the supervisory role constitutes the primary point of contact between the student(s) and the department. It is crucial that this role is figured in such a way as to maximise the benefit for all parties.

Assessment

It is difficult to transfer familiar models of assessment (constructed in relation to small pieces of coursework and exams) to projects. Partly this is because, with projects, not everyone undertakes the same piece of work to the same deadlines, and partly because they produce a wide range of types of “product” all of which can potentially be assessed. Careful thought has to be expended on appropriate, reliable and scalable assessment regimes.

Reflection

Students not only learn by delivering the outputs of a project, they learn from the *process* of doing a project. This is not always transparent to them. Building in opportunities for reflection on the value of the activity they are undertaking can enhance their experience.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



Team or Group projects

When students work in teams to produce a project, some of the existing problems with projectwork are multiplied. Other problems occur only as a result of working with others. Equally some of the benefits of team work are not obvious and not obviously predictable.

Motivation

Project work is an unusual form of teaching in CS in HE (that is to say, it is not the usual lecture-based course); its scale and lack of detailed guidance make motivation an issue. It is important to pay attention to the motivational background of the students and the potential “baggage” they may bring to working on projects.

Each of these sections has been overseen by a different person. Almost always, these individuals are the ones who have worked on the same theme throughout. Each section is structured in the same way, with an introduction to the important issues followed by a set of specific practices which have been prepared for transfer (we call these bundles). More than in any other part of the book, the prefatory pieces outlining the issues are the work of one person, each written in their own “voice”. The style throughout part two is therefore not even, but intentionally so.

What is a Bundle?

A bundle captures a piece of practice that we think is in some way “good”. It might be the best way we have seen for addressing a particular problem, it might be the *only* way we have seen to address a particular problem. In any case, the practices that bundles encapsulate are real, they have all been used in at least one institution. What we have done is to identify the core elements of the practice and put them (“bundled” them) into a format which makes them accessible. From reading a bundle you should (if your context matches the original) be able to adopt the practice easily, and find no “nasty surprises” of scale, scope or applicability; nor that any essential detail of implementation is missing.

Here is what each bundle looks like:

1. Problem Statement

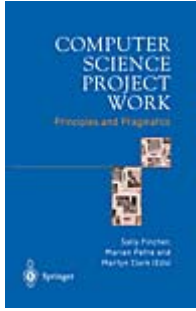
Each bundle starts with a formulation of a *general* problem to which the body of the bundle is a specific solution.

2. Body

The Body of each bundle is presented in a format that shares certain formulaic phrases. These are:

<i>This Bundle</i>	A phrase which captures the essence of the practice
<i>The way it works is</i>	A description of what is involved (this may be quite short, or many paragraphs long. Occasionally it will be many pages, sometimes including detailed documentation.)
<i>It works better if</i>	Key criteria for success

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



- It doesn't work if* Watchpoints for unsuitable (or undesirable) situations
- Every bundle has these. Additionally, they may be supplemented by
- It doesn't work unless* Points which are absolutely required
- You'll know it's worked if* Ways to check that the desired result has been achieved
- Variations* Other ways this might work (mostly, but not always, we have observed these "in real life")

3. Solution Statement

Following the body of the bundle is a *general* solution which refers back to the initial problem statement. (The solution statement, of course, captures the aim of the body too, because a bundle is itself a specific instance of the general solution).

If you read the problem statement and find that it does not apply to you then you can skip the rest if you want. However, if the problem is applicable but the body of the bundle will not fit your context, then the generalised solution statement should tell you what you have to do – if not specifically how to do it.

References

1. Busby, J.S., *Effective Practices in Design Transfer*. Research in Engineering Design, 1998(10): p. 178-188.
2. EPCoS, *Internal evaluation comments*, . 1999.
3. McNeill, W.H., *Diffusion in History*, in *The Transfer and Transformation of Ideas and Material Culture*, P.J. Hugill and D.B. Dickson, Editors. 1988, Texas A&M University Press. p. 75-90.
4. Salisbury, C., *et al.*, *The implementation of evidence-based medicine in general practice prescribing*. British Journal of General Practice, 1998. **48**(437): p. 1849-52.
5. Alexander, C., S. Ishikawa, and M. Silverstein, *A Pattern Language: Towns, Buildings, Constructions*. 1977, New York: Oxford University Press.
6. Fincher, S., *Analysis of Design: an Exploration of Patterns and Pattern Languages for Pedagogy*. Journal of Computers in Mathematics and Science Teaching, 1999. **18**(3): p. 331-346.
7. Alexander, C., *et al.*, *The Production of Houses*. 1985, New York: Oxford University Press.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



Allocation

The allocation process usually results in a many-to-one mapping from students to supervisors; associated with each link is a defined project. There are many ways of conducting this process, but it should be self-evident that as numbers increase, so does complexity. This brings with it concomitant opportunities for error and dissatisfaction.

Allocation is a many-faceted process - it may be construed as:

- the allocation of students to supervisors (who then define a project)
- the allocation of projects to students (with an implicit supervisor), or
- the allocation of a student-supervisor pair to a project

Which of these constructions is applied is an issue of local management, and may operate under any number of personnel and resource constraints. It is not uncommon to see satisfactory mechanisms develop serious tensions over time as staff and student profiles change, resulting in a major change to the allocation mechanism.

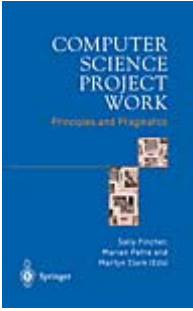
It is Important not only that Justice Is Done, but also that Justice is Seen To Be Done

The consequences of getting allocation wrong, either wholesale or just for unlucky individuals, are usually very bad, particularly for the students. Misallocation (actual or perceived) can be extraordinarily de-motivating, and the consequences of this, given the characteristic weight of projects, are severe. Misallocation can lead to, *inter alia*, misunderstanding of the project, or supervisor, or dislike of the supervisor (or, unprofessionally, supervisor's dislike of the student). The day the allocation list is published - usually an open and public mechanism, well advertised in advance - there is often much distress and complaint. This is probably unavoidable, but it is important that students perceive the allocation process as largely "fair", since this will dilute resentment. There is often an understandable, and sometime accurate, suspicion that staff will cherry pick their "favourites", and all steps should be taken to prevent this happening, and demonstrate publicly that it doesn't happen.

Beware the Beauty Competition

It is natural behaviour among many students, given free will, to queue up in front of the "popular" members of staff, howsoever judged: this "beauty competition" is often going to be at variance with academic best advice. Since it is common practice to spread supervision load across the whole staff, it is highly likely that some supervisors will be quite unknown in name and face to students, which is clearly intimidating. This is especially the case when non-academic staff (Research Fellows, or support

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



staff), or staff in other departments are drafted in to supervise [See 2.12, *Pulling in the Bodies and Graduate Students as Supervisors*]. This important fact is often forgotten.

Equally, many staff will be unaware what drives student choice; it might be expected that the topic, (especially for research focussed staff) is the prime motivator and it will not be perceived that impressions received during teaching experiences in early years count very heavily. At the same time, staff must perceive the allocation as "fair" - for example, allocating five students who are all expected to graduate with third-class degrees to one individual is unlikely to be received enthusiastically except by the most devoted teachers.

Loading and Co-ordination

In most instances, the allocation process is managed by an individual - it is difficult to see an organisationally acceptable way of distributing the core of this task. Like many administrative jobs, this is onerous and will not be popular. The choice of this individual can be critical, since whoever is chosen must have an insight into broader staff loading issues, and must have sufficient weight (or kudos) to be able to persuade or coerce colleagues into doing things that they may well not want to. Normally, this individual will have a senior member of staff as back up for the occasions that issues of seniority interfere with preferred allocations. These can be delicate managerial issues, dependent on local personalities and practices, for which it is difficult to plan.

What a "reasonable" load may be is likely to be a matter for local resource allocation [see 2.12 *Staff Deployment*], but experience suggests that if one individual is simultaneously supervising more than six independent projects, the resulting context-switching begins to disadvantage the students. Some mechanisms exist for streamlining the supervisory task, but when the system starts to creak because of shortage of supervisor time, it is usually time to consider a major rethink.

Special cases may arise in the event of, for example, MSc projects conducted over the summer. These provoke a special tension in that they usually require supervisors to be specialist academic staff (precluding the use of peripheral staff) at the precise time they are accustomed to uninterrupted research and conference time.

Do not forget that the whole allocation process usually has to be repeated, at least in part, in respect of second markers, or shadow supervisors [see 2.9(iv) *Moderator and Supervisor plus Another* and 2.11 *Co-ordination Structures for Supervision*]. Often this is overlooked and done in an unseemly hurry; this is a process that requires matching of staff to staff, with all the concomitant problems of local personality issues.

Especially, remember that you must have a contingency strategy. In less than 1% of cases, something will go wrong and a student/supervisor partnership will have to be changed. You must have a strategy so that you can react in time [see Section 5, *Stuff*

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Happens,]. The sub-issue of allocation of students into teams and groups is considered in Section 8 "Team/Group projects", and particularly 8.2 and 8.3.



Unhappy marriages - topic allocation

Much the most commonly experienced problem is the mismatch of staff to student "demand" areas or, conversely, the mismatch of student desires to topics offered by staff. The latter here may be at the coarse level ("I don't want to do AI") or finer grained, if the local practice is to publish lists of precise project possibilities ("I don't want to do a project on simulated annealing"). This is another aspect of the difficult cost- minimisation task faced by the project co-ordinator or manager, and is intertwined with local expectations: Do students *expect* to be able to work in their chosen area? Do staff *expect* only to have to supervise in their research or cognate areas?

Other Connected Issues That Need Careful Thought Are the Handling of ...

- ... projects created or introduced by the students themselves. These are common, especially in the wake of a sandwich placement or specialist vacation employment, and represent an important opportunity for the student and department to reinforce useful links with outside organisations. Very often, however, student or company expectations are wildly unrealistic, and careful and tactful intervention is needed to re-scale and re-focus ideas.
- ... projects conducted on behalf of "outsiders". These may be other departments within the university or companies entirely independent - they have the same problem of definition as student defined projects, but the secondary problem also of finding a student to take them on. It may be that the client is in some sense "cherished", in which case there may be pressure to allocate a student especially likely to succeed, implying cherry-picking of the talented end of the cohort. This will almost certainly be at variance with issues of fairness.

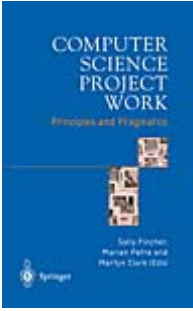
In both circumstances, the issue of first loyalty must be spelled out - that is, the project is being conducted in pursuit of a degree and may result in a useful product, not the other way around.

... Differing Student Ability

Orthogonally, problems often surround strong students taking on projects which do not challenge them, or weak students attempting something just too difficult [see 2.1, *Weighted Topics*].

When it is possible to gauge in advance how tough a project is likely to be, then some care can be taken to prevent this - it is not, of course, always the case that this will be possible. Another aspect of this issue is the *type* of project - many, and often the

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



majority, are of the *design-and-build* kind preferred by accreditors, but examples of pure research projects are not uncommon, and many examples exist of very worthwhile activity conducted under the project head whose connection with computer science may be arguable. Some especial care needs to be taken in matching students to things which are out of the ordinary.

Quality Assurance

Quality assurance in supervision (extracting, or hoping for, uniform treatment from the staff cohort) is a known problem [see section 6, *Supervision*]. If it is practice to use non-academic staff as supervisors this problem becomes more serious since often such people are unaware of the mechanisms mainstream teaching staff take for granted - this is particularly the case if they are pursuing "pet" projects of their own, with the danger of the project becoming solely product focused. This is, in fact, a special case of the general problem of non-uniform experience and expectations among supervisors.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

4.1 Me and my shadow ...

Where projects are double-marked, the second "supervisor" often doesn't see the project until it is completed and handed in

---ooOoo---

This bundle aims to fill any "gaps" that may be expected or anticipated by the allocation of a second member of staff to existing student-supervisor pairs. A secondary aim is to spread the talents of the supervising staff as widely as possible.

The way it works is to ensure that the allocation of shadows is done sufficiently early in the process to have meaning well before the final assessment, and to require some correspondingly early input. This gives an opportunity for tangible inputs both to supervisor and student.

The project co-ordinator (in almost all instances, a single individual) may be expected to have a reasonable idea of the talents of individuals - these include skills of supervision (based on feedback from earlier years, or less formal understandings), experience (contrasting new with more established staff), experience with "how things are done here" (enculturation), as well as technical or specialist expertise. It is not unusual for the allocation process to have projects being supervised by staff who are not specialists in the topic area, and likewise for some students to be supervised by "beginners". The allocation of shadows can then take this into account and ensure that supervisor-student pairings of potential weakness are bolstered by the shadow having the appropriate skills.

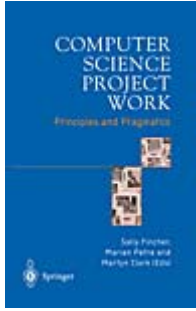
Identification of likely "gaps" can be achieved by stipulating a brief, very early deliverable that itemises the project's objectives and probable skills requirements - this can be the contract between the student(s) primary supervisor. Such a document permits the co-ordinator to see quickly where skills shortages might lie.

Early involvement of the shadow can be achieved by requiring an intermediate deliverable that gives fuller details of the project plan, perhaps with bibliography and progress report [see 7.2 *Mid-project Report*]. This report should be assessed, or at least scrutinised, by the shadow, requiring written feedback that both student and supervisor will see. This can be brief (in the form of "tick-boxes", plus comments). This involves the shadow in partial "ownership" of the project at an early stage.

It is a very good way of getting experienced supervisors to talk directly to the less experienced, and provides a mechanism for ensuring that any "lame ducks" among supervisors are backed by a safe pair of hands. In the same way, inexperienced shadows can be allocated to supervisors known to be reliable and safe, to expose them to good practice.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



It works best when the co-ordinator has a clear understanding of the strengths and weaknesses of the supervising staff, and the authority to make pairings as desired.

It doesn't work if the co-ordinator is short of some of this knowledge, or is not given sufficient information about the projects being conducted, or if affairs are conducted in a rush.

---ooOoo---

So: use shadows for more than double-marking, to plug the gaps that you know are going to occur.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

4.2 “I'd like to do that”

Students often devise projects of their own, but with scant idea of what a project represents.

---ooOoo---

This bundle aims to retain student enthusiasm while ensuring a student-proposed project has a prospect of academic success.

The way it works Students often propose their own project ideas, and in some places these represent the majority of projects. The ideas may come from previous modules, from previous employment or placements, or other connections (“*My Dad's company needs ?*”). These ideas are often ill-formed and would not make successful projects?the reasons may be to do with scale, scope, content, divided loyalty or a number of other possible causes. When ideas are over- (or under-) ambitious, it is important to put them back on the rails without losing student motivation for their own imagination.

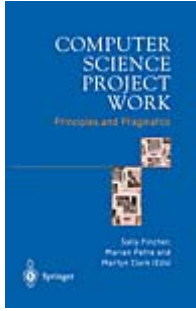
You have to devise a clear proposal format in which student-originated ideas may be presented, and a protocol for their consideration by the department. This should mirror the local project practice as far as possible in respect of deliverables and, in particular, schedule. The undertaking on the part of the department is to make every effort to mould the proposal into something acceptable for proceeding?for example, by recommending changes to priorities or methods.

The benefits of the exercise can be great since the student is required to think well outside issues such as modules and grades in framing a persuasive proposal. Good things to require in a proposal include:

- An indication of the student's desired learning outcomes: "At the end of the project I will be able to ..."
- A specification of curricular material that the project will exploit: If there is none, the project may easily be seen to be inappropriate
- A *student* estimate of how "hard" the project is: For example, will it win a First if well acquitted?
- An indication of staff who This can lead to discussion of what input - technical or academic - the

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>





might supervise:

project actually needs, and where this might most suitably be found. For example, for "external" projects, technical help can be sought outside, but there must be a fair way of accounting for such input.

- Resources, and their costing:

Especially useful if hardware or software not commonly or publicly available is sought

Such points can form the basis of useful negotiation along the lines of "You need to include some ?.", or "If you want a respectable grade, you will need to ?".

It works better when combined with a format which requires intermediate deliverables [see 7.2 *Mid-project Report*], which allow the project to be regularly checkpointed. It works best with a (potential) supervisor who is prepared to negotiate with the proposer.

It doesn't work if staff are unwilling. This is a real problem for many such proposals, particularly if they originate in a commercial domain removed from local interests. It will also founder if the primary loyalty of the student is not to the academic outcome?that is, if the project is seen primarily as a product generation exercise for some external organisation.

See also: 2.1, *Externally-provided (or negotiated) topics*

---ooOoo---

So: provide an explicit project framework to scaffold student enthusiasm and imagination, and use it as a basis for negotiation and contracting.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

4.3 Project sabbaticals

Like everything else in the academic cycle, projects can become a treadmill. Projects usually consume more than a term or semester (often well over half the academic year), and in most departments "there is no escape". It is important for staff to be able to get some relief from this treadmill.



---ooOoo---

This bundle proposes periodic relief from some or all supervisory duties. The benefits for the staff are in an opportunity to recharge batteries, and for students in consequent renewed enthusiasm.

The way it works is for the department's workload allocation to recognise project supervision very explicitly and permit buyout from all or part of it. The precise "price" that might be paid is a matter for local definition, but obvious options are: responsibility for all or part of some other teaching, or a suitable administrative task. It is not necessary for *all* supervision to be excused - a half-load can have a surprising rejuvenating effect on supervisors who have been responsible for, say, 6 projects a year for a decade.

It is not necessary for this to be part of a larger sabbatical arrangement; it is more in the way of a local rearrangement of duties.

It only works (obviously) if the department's managerial hierarchy are persuaded of the merits of the idea, and are prepared to see appropriate rearrangements of duties.

---ooOoo---

So: provide mechanisms to give people a temporary break – they will come back to the task better motivated and with more energy.

4.4 Dynamic Matchmaking

"Free-for-all" allocation mechanisms which involve students (and/or staff) selecting projects themselves are always time-consuming and often unsatisfactory for a substantial proportion of those involved.

---ooOoo---

This bundle provides a mechanism for establishing student-supervisor pairings in an environment of large numbers of staff and students?probably unknown to each other?in which there is a possible skills-and-interest mis-match.

The way it works is to partition the *topic areas* that projects occupy as uniformly as possible (for example, AI, graphics, theory ?) - this partition is of necessity coloured by local supply and demand, but may be amended each year on the basis of demand experienced the year before, and adjustments to the supervisors' profile. In most cases, the assignment of supervisors to areas is quite easy to determine, although it is probable that some will be more adaptable than others.

Students are invited to nominate three (say) topic areas in which they are prepared to work - these would normally be prioritised. Where pre-specified projects exist or have been negotiated, they will normally belong to one of these categories and a preference for it may be indicated by the student.

At this stage the complexity of the allocation problem is much reduced and it is a feasible task to construct by hand a supervisor-student pairing by matching on preferred topic areas. It is important that this is done by a single individual who has an overview of all the issues and personnel involved, and in whom both students and supervisors have trust. It is possible that software could ease this task, but it is likely that knowledge of local personalities (either student or staff) would always be needed to fine-tune the allocation. The allocation may then be used as the basis of negotiation toward a full project specification - a nice feature is that the final project may potentially be nothing to do with the originally nominated topic areas, provided supervisor and student agree.

Given that there are staff to cover enough topics (not a problem in a moderate-sized community) this mechanism can remove the "beauty competition" aspect of allocation by making it difficult for students to pursue individual staff as supervisors.

It works better if the match of staff and student preferences coincides well with the topic areas defined. There is an obvious element of serendipity in this, but some preparation can be done by ensuring that all reference to the project in earlier years is in the vocabulary of the topic split preferred.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

It doesn't work if significant numbers of students are unwilling or unable to complete forms (in which case their allocation has to be random), or if significant numbers of supervisors are very restrictive about the types of projects with which they are prepared to become involved. It is also necessary that all parties abide by the allocator's decisions.

---ooOoo---

So: ensure allocation is under the control of a single individual, and give them a mechanism to make the task manageable.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

4.5 Musical chairs

If there are many students interested in one project, all but one will be losers.

---ooOoo---

This bundle provides a mechanism for open and fair allocation of highly popular projects that has the side effect of preparing students for the job application process.

The way it works It is to everyone's benefit (staff, students, commissioning companies) to ensure the student best suited to any given project is the one allocated ("best suited" might include extra-curricula skills). In a competitive situation it is worth taking time to make that match.

The primary aim of this bundle is to optimise the match of skills to project, but its use also reduces discontent among disappointed students.

What you do is to use a competitive allocation mechanism for the projects that are heavily over-subscribed. Interested students are invited to regard the project as a job, and to submit a formal application.

Actual allocation may be done on the basis of the written applications or, if time and resources permit, by interview. The latter approach is to be preferred, both to extend the experience and to provide a more open mechanism. It is particularly useful if (where appropriate) employer representatives can be included in such an interview panel, both for the quality of the experience and because the employer then takes a responsibility, and consequent interest, in the identity of the student.

It doesn't work if the selection procedure is not detected by the students as properly "fair" - for example if it is rushed, or conducted behind closed doors. Feedback to unsuccessful applicants can alleviate this.

It doesn't work unless you can devote sufficient time.

---ooOoo---

So: solve a problem of over-subscription by exercising skills which replicate allocation in the "real world"

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

4.6 Horses for Courses

Often a department will offer several types of project, but put them all through a single one-size-fits-all allocation mechanism

---ooOoo---

This bundle proposes that you use different (and differently appropriate) allocation mechanisms for different types of project.

The way it works is that you distinguish between "Research" and "Design-and-Build" etc [See 1.5 and 1.6] with regard to the allocation process. If a research-type project is offered, then the allocation process should be along research (or research-group) lines. Ways in which this can be implemented are: writing a formal proposal which is reviewed; an interview with the head of the appropriate research group; a personal recommendation from a research supervisor. These methods are clearly inappropriate for "Design-and-Build" projects and should not be used for them.

It doesn't work if you only offer one type of project, or if your allocation mechanism is defined (by QA or other stipulations). It doesn't work if the types of project you offer are already confused (on paper, in presentation or in the minds of students, or staff).

It works better if the "type" of the project is clearly distinguished in advance, so that students know what the difference is, and that the difference starts with allocation.

---ooOoo---

So: use appropriate instruments for allocation



4.7 Job application

It is unnecessarily artificial to allocate students to projects on a "skills-blind" basis.

---ooOoo---

This bundle provides a mechanism for allocating students to the best job for them, using a job application process.

The way it works is that students have to fill out job applications before entering the course (in its original form this was a course which ran over three-quarters of a year). If their application is accepted, they may register for the course and start out as a fledgling worker; they can potentially work their way up to project manager. As they work, they focus on a specific portion of the project.

Even over three-quarters of a year, they may not see the software to completion, so others continue the project after them. In order for others to continue, they need the thinking and reasoning of past team members recorded in the form of requirements, design, software, and test documents. These documents mature as the project progresses, but cannot be rewritten every term or progress on the project will not be maintained.

It doesn't work if your project is constrained (by Professional Body requirements, or the like) so that every student must experience every part of the project life-cycle.

---ooOoo---

So: respect and reflect the differences in student ability in the same way that industry does.

References: This bundle is based on original practice developed in *Real-World Lab*, started by Melody Moore when she was at Georgia Tech. A variant was reported in the FASE newsletter (*Forum for Advancing Software engineering Education (FASE)*) Volume 9 Number 08 (115th Issue) - August 15, 1999) by Susan Mengel of Texas Tech and further details can be found at: <http://www.se.cs.ttu.edu/>

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Supervision



As student numbers have increased, many of the traditional forms of individual contact between staff and students have declined. Regular and continuing "tutorials" seem to have been largely abandoned in all but the best-funded Universities; pressure on staff time has seriously curtailed availability for casual enquiries on an "open-door" basis, and led to the uptake of "office hours". Today, one-to-one contact most often occurs in the context of project supervision, greatly adding to the significance of this role in the students' development.

Project supervision is an activity that encompasses several aspects, including:

Technical Assistance

Because project students typically rely (in the first instance) on their supervisor for detailed technical assistance it is most common that supervisors take on projects in their areas of expertise. There are several allocation mechanisms [see, for example, 2.1, 2.2. and 2.3] that address this. The driver for providing this sort of assistance is most often demand from the supervisee(s) and the supervisory input is reactive.

Engineering Processes

Especially in large scale (or group) projects there is an increasing reliance on effective use of engineering processes by the students. Sometimes processes are prescribed (typically where all students are doing the same project) but where this is not the case, they are often required to choose appropriate processes for the demands of their task. The driver for giving this form of assistance is often failure (or imminent failure) of the students' process, spotted by the supervisor: the supervisory input is proactive.

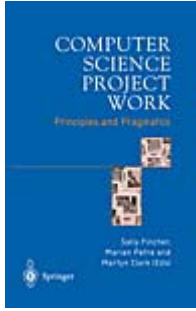
Personal and Professional Development

Many of the problems encountered by students during large-scale and long-term pieces of work are connected with external circumstances rather than the work itself. These may be concerned with the difficulties of working within a group, with legal, ethical or property rights, or with the students' real life. The drivers for giving personal or professional development assistance are as often concerns raised by other stakeholders as they are problems observed in (or by) the student immediately involved.

These aspects are rarely distinct or discrete, but are all part of the activity of supervising a project and must be taken into account.

Where more than one member of staff supervises projects, look for inconsistency among them.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



Many project instances assume that there is a consensus amongst supervisors as to the aims and objectives of the instance, and a consistency of approach between supervisors in addressing those objectives. This can, indeed be the case among a stable group of supervisors where such a consensus has been developed (maybe implicitly) over time, but the privacy of the supervisor-student relationship makes such a consistency difficult to demonstrate. In any event, new supervisors being bought into such a culture will not have the benefit of this history, and must (at least) be indoctrinated into the zeitgeist.

An apparent collective understanding may, without having a noticeable effect on assessment outcomes, conceal wide disparities in affective goals (of supervisors as well as students). Consequently, either:

Formulate agreed goals for the project instance.

This is in keeping with the current climate of specification of expected student learning outcomes and experience, but can be difficult to assure across a large pool of supervisors.

Factor the disparate approaches of supervisors into the assessment criteria used

This can be easier to implement if the supervisory load is shared (as it often is) by a wide variety of staff, but may require an additional step in the assessment process to account for the supervisor's contribution to the students' achievements, as well as recognition of the potential variability of that input.

Use a smaller pool of supervisors, and expend more effort on entraining them in forming a consensus.

This approach has been used successfully even with single supervisors taking on quite large cohorts, but requires dedication from the staff involved, and co-operation from those apportioning teaching loads. Unless care is taken, it can also lead to a decrease in the types of projects offered, to the detriment of the ability of students to express their enthusiasms.

Cost

Supervision is an expensive activity, both at a departmental and individual level. There are two principal ways to address this. One is to maximise the use of currently available resource (staff time), exemplified by bundle 5.2 *Loosely co-ordinated groups*; another is to supplement, augment or otherwise increase the scarce resource. An example of this is given in 5.4, *The Supervisor's Eyes and Ears*.

Projects are (usually) conducted over a long period of time for large credit. There is a need to intercept failure early

At one extreme, it is possible to construe project supervision as simply a "fly on the wall" activity, with students being allowed to follow their own paths, even to the

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

extent of being unable to deliver anything at the end of the day. This is rarely taken to be an effective learning process. Thus it is concomitant on the supervisor to attempt to detect such pathological behaviour, and to help the student to change it before it is too late. At the other extreme, supervisors can control students' behaviour to the extent that students have no creative (high-level) input to the work they are doing and are effectively unable to fail.

It is particularly difficult to keep students on track if there is no track. The first task of a supervisor is thus to ensure that students make (or buy, or copy) a map. The second task is to help them follow it - to an appropriate extent. The simple existence of a map (whether a project plan, or a series of interim deliverables) does not imply that students should always be penalised for deviation (there may be a better way), but does provide a common reference against which student and supervisor can reflect on and discuss progress. Such intermediate supervisory input can be useful in affecting working practices, whether it is to alter the bad or assist in the good.



Conflict between supervisor/assessor role

Where a supervisor is also responsible for assessing a project, there are inevitable tensions between the two roles. Students may perceive that advice given by the supervisor is coloured by their separate duty to assess them. So they may form the perception that the supervisor is withholding (or giving) particular advice not to support and develop the students' own thinking, but as part of their role in assessment.

There may, in fact, be a conflict between what students should do in order to complete a project in its own terms, and what they should do in order to maximise the mark awarded. In an ideal world, with assessment strategies completely aligned with the work undertaken, this would not be the case, but projects are often deliberately artificial in their nature. For instance, it is often the case that quite small projects are used to give students practice in the deployment of software engineering processes only applicable to far larger instances. In this case, the rational supervisor would advise students that it was better to use much simpler processes in the work, but assessment objectives dictate otherwise.

In institutions where examinations are marked "blind", projects are probably the most significant pieces of work undertaken by students in which their identity is visible. Indeed, blind marking of exams is often adopted precisely to combat the unwanted effects of the relationship between students and staff that is claimed as a major benefit of the project experience.

Stuff happens

Things always happen. They are always unexpected. This section has been compiled from anecdote, "war-stories" and bar-room discussions with many colleagues. None of the problems presented here may ever happen to you, but something will. The

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

important thing is to retain the expectation that there will be an exceptional circumstance somewhere and to have (even half-formed) contingency strategies.

The supervisor is not expert in the subject area or in regard to student process.

The mix of projects which students are interested in undertaking often does not match the mix of staff available to supervise them. Research-oriented staff may have a partial understanding of the requirements of process-based projects. [See also 4.6, *Horses for Courses*].

Conflation of student problems

. Some students will have problems in areas other than their project work. Equally, students (particularly "traditional" 18-year old students) are not very good at separating areas of work and their emotional reactions to them. Given the personal nature of the supervisory process, these will often impact on projectwork - where they don't really belong [see also section nine *Motivation*].

Student time management - conflicting priorities

Projectwork, especially undertaken in groups, can consume far more time than its assessment-worth. Be prepared to make them work less on the project - even if it is your pet idea.

Late breaking failure

Students are good at covering up lack of progress, both their own and that of other non-functioning members of their group. Potential failures discovered near the end of a large project are far more difficult to retrieve.

Ducking for the tape

Where projects finish near the end of an assessment period there is no scope for extension. This is an artificial situation leaving students having to make a decision about what to omit from their delivered work. This will be conditioned by assessment criteria rather than by software engineering realities.

Project turns out too simple/complex

Especially where students specify their own projects it can be difficult to judge an appropriate and comparable level of complexity.

Group geography

Students who live together interact differently. So do students who live far apart.

Role allocation - star and forced-drone

In a group situation where students allocate their own roles, as well as the familiar reliance on the skills of one or two members there is a potential problem with a competent student being assigned (or taking on) a role which does not allow them to demonstrate their abilities.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

5.1 Characterising Supervisor Input

Variations in the student/supervisor relationship have an effect on the extent to which the project is "the students' own work". These are (commonly) not visible, and therefore not factored into assessment.

---ooOoo---

This bundle provides a mechanism by which the level and scope of supervisor assistance can be recorded and factored into assessment, and coincidentally provides a framework in which supervisors can reflect on their own approaches and effectiveness.

The way it works is that, as part of the assessment process, each supervisor is required to report (on a pre-printed form) the extent to which they have assisted students during the course of the project. In order to facilitate comparability, the supervisors are asked to characterise their role as (predominantly) one of four types:

Observer/commentator

The student(s) run the project for themselves, sometimes in the presence of the supervisor. The supervisor gleans from their actions the roles that they have taken, and the success with which they are implementing these roles. The supervisor may suggest lines of attack or potential solutions to problems, but will not in general require that the students adopt them.

Master/mentor.

The traditional "apprentice master" role with the supervisor passing on skills and advising the student(s) on the approaches to be taken to specific tasks.

Line manager.

The student(s) manage and implement the project, but regularly report to the supervisor who tracks progress and dictates strategy but not tactics.

Project manager.

This is the traditional "science" model of project supervision, where (typically) a small aspect of the supervisor's research project or interest is marked off as the students' project, to be integrated on completion. The supervisor has a particular interest in the success of the students' project, and will guide them to that end, allocating sub-tasks and directing progress.

The supervisor's report is attached to the project before it is forwarded to the moderator, second or other assessor.

It works better if staff share an understanding of the contents and import of the supervisory report, both as authors (supervisors) and readers (assessors).



It doesn't work unless projects are supervised by different members of staff.

It doesn't work if the supervisor alone assesses the project (although there may be a personal development gain for staff in the process of reflection).

---ooOoo---

So: moderate, or at least make explicit, the amount and type of assistance being given to project students by their supervisor.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

5.2 Loosely Co-ordinated Groups

Supervisor time is expensive and may be squandered by duplicating advice. It is important to maximise supervisory resources without impairing the quality of the interaction.

---ooOoo---

In this bundle group projects are supervised not by meetings of individual groups with their supervisor, but by more formal meetings of multiple groups with a single staff member present.

The way it works is that overall staff time spent on supervising projects should reduce, time spent responding to technical (or other) queries outside the formal supervision period should not increase and students' requirements for advice/guidance outside scheduled sessions should reduce.

At each meeting, one representative of each group present makes a short report on the group's progress during the preceding period and its plans for near future. All present respond to this presentation and to any problems raised.

Meetings last one hour, with each group's presentation lasting 10-15 minutes (including Q&A/problem solving). Students take the role of Chair and Secretary for these meetings (on a rotating basis), with minutes and paper versions of the progress reports being circulated to all after the meeting.

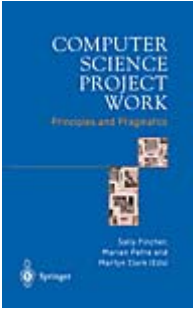
Using this bundle makes staff/student interaction more focussed, and hence more useful to both parties. It gives students the experience of chairing meetings and of taking minutes, gives them practice in the presentation of results outside formally assessed forums and introduces the idea of group monitoring, additionally it encourages students' responsibility to the cohort as a whole.

This means that staff time is saved by combining the function of similar meetings, students learn to support each other's work and working practices and the amount of input a student gets from a supervisor is made more visible.

It works better if each group undertakes a different project, or each group undertaking the same project is placed in a different loose group (avoiding cross-fertilisation between instances of the same project). It works better if projects are similar enough that the students have some understanding of the work being undertaken, and the technical problems encountered in, other projects. It also works better for group projects, as meetings consisting of individuals each presenting their own projects would either run too long, or be too small to encourage interaction.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



It doesn't work if students' preparation for the meeting consumes more effort than the benefits gained or if students do no work (despite warnings) and are absent from the meetings. It doesn't work if students come from different programmes, or are undertaking projects with a different weighting; leading to a proportionately different amount of effort being spent on preparation for the meeting.

---ooOoo---

So: appreciate that supervision does not have to be a one-to-one activity

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

5.3 The help they've had along the way

At the end of a project, only the deliverables are visible. It is difficult to gauge the amount or source of supplementary "help" or "technical assistance" that students have had.

---ooOoo---

This bundle allows staff to record, and supervisors to quantify, the amount and extent of help given to students, and to factor this into the assessment process.

The way it works is that students undertaking a project are given a set number of "vouchers", which they can trade in for assistance from any qualified member of staff. The voucher is filled in by the staff member to indicate the purpose and extent of the help given (and to whom) and returned to the project supervisor for consideration in the assessment process. Thus every submitted project has an associated file of completed help vouchers (which will range from none to maximum).

Each piece of advice or assistance is worth one or more vouchers. The number of vouchers per student may be fixed at the outset, and no more issued.

A Voucher might consist of:

<h1>HELP VOUCHER</h1>	
Group/Student Name	
Help Requested	
Help Given	
Signed:	Date

By using this bundle, students can approach staff who have the required expertise, rather than being restricted to their project supervisor. Students know how much help they can expect with their work and at the same time an upper bound is placed on the commitment of staff to supporting that work (which, in extremis, can be factored into staff loading calculations). Records of what help students have required can be used to identify curriculum "weak spots".

It works better if there is a shared understanding of voucher currency (the voucher/advice exchange rate) amongst staff. It is particularly suitable in situations where all students undertake the same project (in order for comparison of the amount of help needed to be meaningful).

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



It doesn't work unless the market for help is accurately judged. If too few vouchers are handed out students will not get assistance they need and/or a black market may emerge; if too many vouchers are handed out you risk a flood of "use it or lose it" queries

---ooOoo---

So: look for a mechanism which can record the formal (and semi-formal) help the students have received.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

5.4 The Supervisor's Eyes and Ears

Managing group processes can absorb a lot of supervisor time

---ooOoo---

This bundle reduces the amount of supervisor time spent in managing group processes by using postgraduate students in an intermediate supervisory role.

The way it works is that each project group is allocated a postgraduate "advisor". These advisors give no technical input, but focus on group processes and the early identification of problems. Advisors are specifically told that their responsibility is to spot problems rather than fix them. The advisors are paid, so they turn up to meetings and send in reports much more reliably than academic staff. Students meet with their advisor on a weekly basis; the agenda and minutes are included in the student's project log, which is an assessed deliverable (assessment of this is undertaken jointly between advisor and supervisor). An additional assessment benefit is that the advisor can provide independent evidence to moderate or justify extreme mark distributions within the group. Each advisor sends the supervising member of staff a weekly report of their group's progress (which can provide early identification and evidence of non-performance so defaulters can be chased) making the advisor the supervisor's "eyes and ears" on the group.

When problems are identified, the supervisor talks it over with the advisor and agrees on the next step. This may take the form of ideas that the advisor can feed to the group, or it may involve the supervisor talking to some or all the group members. In extremis, the supervisor may sit in on a progress meeting or formally summon the group to a meeting with them.

It works better if the project is of sufficient length and complexity to require investment in process. It is most useful where the students have not worked together before and the groups are mixed or lacking in terms of group-working experience: ideal for second years.

It doesn't work if advisors act as project manager and therefore remove the need for the group to own their process. It doesn't work if they start to give technical advice - which is perceived as unfair by some students. It doesn't work if there are insufficient postgraduates available - in any case new advisors need to be recruited and briefed each year. There can be "advisor" problems with postgraduates who are not experienced enough to spot a group which blows up late in the project.

---ooOoo---

So: see who else might be able to supervise the non-technical aspects of group projects.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

5.5 Looking for the early wobble

---ooOoo---

Projects are (usually) conducted over a long period of time and for significant credit. Without mechanisms for tracking progress, there is much scope for things to go wrong - with dire consequences.

This bundle assists students in producing final project reports by requiring them to submit well specified sections of the report at set times during the year, for example after the analysis, design and testing phases.

The way it works is that students submit reports on specific (perhaps negotiated) phases of their work at the end of that phase, rather than at the end of the project. Thus they can be given feedback on their report writing and have a chance to revise their work before it is finally assessed. Students can't leave all the report writing until the end of the project and then be faced with the horrors of a blank sheet of paper. Conversely, if there is a major revision of the aims or requirements late in the project, the fact that earlier work has already been "validated" discourages students from perceiving it as wasted.

A variation of "stage by stage" is to link the staging to interim grades as well. A way to implement this, in situations where everyone does the same project, is to require weekly progress and then to provide "model solutions" at the end of each week so that everyone starts the following week at the same point and no-one's marks are disadvantaged cumulatively. [An example of this is described in: A. G. Sartori-Angus (University of Natal, Durban, S Africa) Object-Oriented Design through Ray-tracing, Proceedings of the 5th Annual Conference on the Teaching of Computing (26 - 29 August 1997) pp 220-222]

It works better with final year, individual projects where the development of the project can be expected to show definite phases and require a substantial written report upon which much of the assessment will be based. **It doesn't work unless** the requirement for staged delivery is common to all students and supervisors. The detailed content of the staged reports can be negotiated between the student and supervisor to be appropriate to the project's development method. Project supervisors must have sufficient time to give the necessary feedback; if they don't students may be resentful of the supervisor not keeping their side of an implicit bargain.

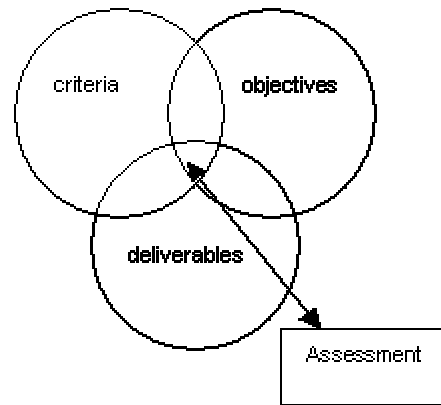
---ooOoo---

So: institute mechanisms which trap early failure

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Assessment

Assessment sits at the intersection of learning objectives, the deliverables which demonstrate those objectives and the criteria used to judge the quality of the deliverables:



This relationship is not unique to assessing projectwork, it is common to all assessment. However, there are issues specific to projectwork which change these aspects.

Projectwork is, in general, at a larger scale than other assessed work; indeed, this is part of the rationale for having students undertake projects at all. Many of the problems of software development only become apparent as scale increases. Small-scale coursework typically consists of implementation from a given design, or design from a given specification, where there is neither scope for innovation nor necessity for multiple deliverables. This increase in scale also leads to a change in the nature of the required deliverables ? large scale projects inevitably requires a more rigorous approach to the process of generation of the products, which process must itself produce assessable artefacts. This is congruent with the principles of software engineering.

In many cases (particularly with final year projects), students are undertaking significantly different projects from one another, leading to a disparity in appropriate deliverables which must still be assessed under the same "rules". This can also make it especially difficult to generate criteria which are precise enough to be meaningful as aids to assessment, yet general enough to cover the range of projects undertaken in a particular instance.

In team and group projects "working together" is often an objective, and is often the first time that such methods have been required. This forces a change in the assessment criteria (especially in the minds of the students) where previously deprecated practices of producing joint work ("copying", "plagiarism") become an essential, assessed component.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Ensure that the objectives, deliverables and criteria are in constructive alignment

If one of the objectives of a projectwork instance is that students learn process as well as produce products, then it is important that the deliverables of the project actually demonstrate stages and aspects of that process. [See: 6.5 *What is a "report", anyway*].

Marks are the only thing we've got that students want

~~Most~~ Some students are not motivated by the Joy of Learning. They are at University for the benefits that the qualification will give them in the job market, and therefore work primarily to maximise their marks. Staff can use this to their advantage in the assessment process by setting criteria and deliverables that manipulate students into learning the important bits, by giving them the most marks. [See: 6.4 *"Authentic" Assessment Criteria*].

Assessing projects is an expensive activity

Partly because of the scale of projects, partly because of the weighting they carry in the curriculum (from 3% to 30% of an undergraduate degree), partly because there are often a great variety of deliverables in a great variety of media (software artefacts, technical documentation, reports, code, oral presentations etc.) and partly because several members of staff are commonly involved, assessment of projects is a time consuming (and therefore expensive) activity.

~~Most~~ Some staff perceive the extra work that projectwork assessment requires as excessive and unfair. One way to address this is to undertake an audit to see how long staff actually *do* spend reading reports/watching demonstrations etc. This might prove that project assessment is both costly (it takes a lot of time) *and* cost-effective (it produces results which could not be gained in a different/cheaper way).

Conflict between Supervisor & Assessor Role

Because project assessors are frequently the same as project supervisors, assessment may be coloured by their knowledge of students' effort. If they've "come a long way" should they get as many/more marks than another project which went further, but was easier? It is important to forge a consensus amongst supervisors and examiners as to the relative value of these two aspects.

It is important for the supervisor to have knowledge of where the "tricky bits" of the project fall so that apparently trivial pieces of implementation can be properly acknowledged, but it is equally important to have criteria that distinguish between effort and achievement.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Relationship of Individual Marks to Group Marks

There is a tension between assessing the products of a group and rewarding the efforts of individuals within that group, between "Group products get a group mark" (every group member gets the same mark, regardless of contribution) and "Only individual effort is rewarded" (attempting to completely determine individual contributions).

Although tempting on grounds of economy of assessment effort, "group" marking can be perceived by students as unfair, but is defended by its adherents on the grounds that if individual marks can be accrued from group work, then students will work to maximise their own mark.

Mechanisms which reward individual effort in a group context are confounded by the difficulties which assessors have in determining what actually went on in the group, and by problems of determining appropriate weightings for different activities (e.g. "implementation" versus "documentation").

Between these two extremes, a proportion of the marks can be awarded for individual contribution to the work, with the rest going to the group as a whole. Many mechanisms have been devised to address this; commonly a proportion of the marks is handed over to the students to allocate amongst themselves so that they can reward the highest contributors (and punish the laggards).



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.1 Use Peer Assessment

Especially in projects that are returned to students near the end of their programme of study, feedback tends to be ignored, and notice is only taken of the mark awarded.

---ooOoo---

This bundle engages students in the process of assessing their own work against given criteria and generating feedback on the work of others against the same criteria. It encourages them to see their work with "others' eyes", appreciating their own strengths and weaknesses, and to evaluate the work of others in a domain (and against criteria) with which they are already familiar.

The way it works is that the project is developed to a set of detailed criteria (effectively a marking scheme). This focuses the students, from the outset on how the work is going to be assessed. When they submit their work, students attach a self-evaluation. They then undertake a peer-evaluation of the work of another group who have done the same project.

Students peer-assessing a project do not receive the self-assessment which goes with it, although they will, of course, be informed by the experience of their own self-assessment.

The final mark awarded for a project is decided by the staff member involved, using the self-assessment to moderate the peer-assessment, and referring to the original project (i.e. re-marking) in cases of gross disagreement. The results of the moderation are included with the peer-assessment, thus the students receive the comments of their peers and of the "trusted" staff member.

It works better if students have some experience of peer-assessment, and if the projects are the same or very similar in nature.

It doesn't work unless assessment criteria (including marking scales) are well understood and shared before the assessment process is begun.

---ooOoo---

So: get students to use assessment as part of their learning

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.2 Assessment Walkthrough

Especially where students are working in groups, or attempting the same or similar projects, it can be difficult to decide *who-really-did-what*.

---ooOoo---

This bundle allows staff to validate that the work described/delivered is that of the student, to ensure that the claims made for the deliverables are correct, and to clarify any statements made in the final report.

The way it works is: students meet with staff to undertake a terminal "walk through" of their project deliverables, using any of the standard forms described in the software engineering literature, depending on the goals of the project as well as on how many students are involved. Possibilities include: review of designs against specifications, implementations against designs, code against local standards. (Pressman, R. S. (1992) *Software Engineering, a practitioner's approach*, McGraw-Hill, New York.). It may take the form of a presentation (although there is still the possibility of students "hiding" here) or a dialogue. The results of this process inform the assessment of the students work, but are not necessarily themselves assessed.

It works better if students have had practice with, or are at least aware of, the form and function of software reviews, and hence in projects with a Software Engineering focus.

It doesn't work unless staff and students can commit the time to make it a collaborative, non-intimidating, process.

---ooOoo---

So: use processes derived from software engineering to enhance assessment.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.3 Increase the Granularity

The reliability of assessment suffers when the value of the marginal (unit) mark is small.

---ooOoo---

This bundle focuses student (and staff) attention on the extent of the difference in achievement that is reflected by a single mark - and changes the way assessors award marks so as to ensure the scale rises in appropriate steps.

The way it works is, if each category in the marking scheme (e.g. design, implementation, evaluation) has an associated maximum mark, and within each category marks in the range 0-100 are awarded then it is very difficult to achieve numerical agreement between one assessor's 57% and another's 52%. However, if the granularity is increased, to reflect the marks "earned" by each component (i.e. if Design carries 40 per cent of the total project marks, mark it out of forty instead of 100). Agreement between assessors on what 15/40 means is more likely, and the award of extra marks on appeal from the student becomes more meaningful.

It works better if there is agreement among the staff that granularity is a problem and if agreement about what the weightings should be has been secured.

It doesn't work unless you have a number of categories in the mark scheme.

Variation Students think that 1% on an assessment that counts for of 25% of 1/12th of their degree is worth arguing over. It isn't. Don't argue, but give it away freely.

---ooOoo---

So: make the marginal mark worth something

6.4 "Authentic" Assessment Criteria

If you set a problem that involves programming, students will construct it as being solely about programming.

---ooOoo---

This bundle uses students' understanding of project lifecycles to characterise appropriate deliverables, and to weight the effort (and risk) involved in producing each one. The resulting list is then used to generate the assessment criteria (and associated weightings) for the deliverables of an assessed software project.

The way it works is that, before students start on a project (but after they know what the project will be), a "negotiation" session is undertaken in which the appropriate activities for such a project are discussed, listed and relative weightings are agreed. This discussion can be "steered" and the outcome "stitched-up" by the facilitator deprecating (and discarding) some suggestions and coalescing others, as well as making sure that all the necessary elements are present. This session can be undertaken with the entire cohort (if it's not *too* large) or in smaller groups with the facilitator combining and moderating the results.

The outcome of this should be a list of the activities, a weight for each activity (typically 5-15% of the total effort), and a list of where in the deliverables of the project evidence for the quality of that aspect might be found. This list can then be used to guide students' efforts and priorities in undertaking the project, as (they are told at the end of the session) the effort weightings are used as assessment weightings.

It works better if it is embedded in a software engineering module or programme, so that discussions of weightings are guided by an understanding of appropriate processes.

It doesn't work unless the project to be undertaken is process-centred, and the students already have an understanding of what this means. All students must be undertaking the same project, unless you are prepared to negotiate with each student.

---ooOoo---

So: use pre-specified, agreed and published criteria to direct students' activities in assessed work.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.5 What is a "report" anyway?

The "report", which is the sole assessed deliverable of many projects, is often a mixture of technical and process documentation, and of reflection on achievement.

---ooOoo---

This bundle reconciles the deliverables of a project with its learning goals, minimising the extra work which students have to undertake purely to make their efforts assessable. It recognises that a project is a learning exercise, not a production effort. It also minimises duplication of material in the deliverables that assessors need to consider.

The way it works is: instead of characterising the outcomes of a project as "a report", they are divided into two, separately delivered, sections:

What the project should have delivered anyway, e.g. plans, records of investigations undertaken (requirements gathering), design documentation (including rationales for the choices made), and so on. A reflective piece giving the student(s) perception of the success (or otherwise) of the work, and demonstrating what has been learnt from the process. These deliverables are more appropriate, especially to projects with a substantial software engineering focus, and encourage students to discuss what they have done, rather than rehashing general descriptions of approaches (such as lifecycles) that they have adopted.

They encourage students to submit documents describing what they intend to do (as part of plans and risk analyses) rather than a narrative description of what they eventually did.

The more useful aspects of the "report" ? students' reflection on what they have achieved and how ? is still captured (in the *reflective piece*), but does not now have to be distilled from amongst the narrative description of their software development process.

It works better if the project is not entirely assessed against the final product.

It doesn't work unless there is a consensus among supervisors as to the appropriate deliverables from a project.

---ooOoo---

So: ensure that the assessed deliverables of a project match the process which students are expected to employ.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.6 Assess the fact that they did it

When students undertake a project for an external "client" there are problems with gauging and assessing the scope and difficulty of their work .

---ooOoo---

This bundle identifies a series of tasks which, if completed, will count for the academic credit of the project, irrespective of what the student produces for the "client".

The way it works is that when the project is negotiated with the external (usually industrial) client, a series of tasks are identified whose completion will count for the academic credit of the project, irrespective of the marginal quality of the artefacts they deliver.

These might be in the form of a "learning contract" where the student has to ensure they undertake certain tasks to fulfil the educational objectives of the project, or they might be set in the context of an existing framework (such as the British Computer Society's Professional Development Scheme, see: <http://www.bcs.org/pds>). The fact that the student actually accomplishes the specified activities is certified by someone outside of the academic department (normally in the workplace).

It is assumed that the process and product of the work is satisfactory unless otherwise stated (by the external assessor). If satisfactory, it is sufficient that they have done the work; there is not enough added value in the detail to warrant a more fine-grained assessment process.

It doesn't work unless reliance is placed on the motivation and quality of the participants?students, academic supervisors and external (industrial) assessors.

---ooOoo---

So: consider awarding academic credit for successful accomplishment of tasks, rather than assessing the products of those tasks.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics* , Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.7 Three Wise Monkeys

No matter how tightly specified the criteria for assessing a project, assessors will always find room to vary.

---ooOoo---

This bundle reduces the total variance between assessors by reducing the number of assessors.

The way it works is you only have three assessors per project instance. They grade every project produced within a course (or module or cohort). In this way, they can then spend more time discussing the "implicit" criteria of the project, and are more likely to be able to form a consensus. Their grades will, probably, be closer and internally consistent.

It works better if there is a slow "turn-over" of assessors; that is to say only one rotates out of the job every year. This ensures that the assessment culture of projectwork within the department (the "way things are done here") is maintained and disseminated. It also helps to solve the problem of longitudinal variation in assessment over time.

It doesn't work unless assessors receive "credit" for the work. It doesn't work unless the projects are such that all assessors are equally capable of assessing the technical content.

---ooOoo---

So: consider reducing the number of people who assess projects.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.8 Assessing something is not the same as measuring something

No matter how tightly specified the criteria for assessing a project, assessors will always find room to vary. One assessor's 58% will be another assessor's 49%

---ooOoo---

This bundle addresses assessor inconsistency by de-coupling numeric grades from more qualitative assessments of the work. This provides justification for (and, it is to be hoped, a basis for agreement on) the grade awarded.

The way it works is that everyone who marks the project completes a form which asks the following questions, justifying the numeric mark(s) awarded. An example is:

Mark whichever box best describes the characteristics of the project report.				
The report shows that the student used the available literature ...				
not at all	ineffectively	adequately	effectively	very effectively
The report show that the student's overall grasp of the subject was ...				
minimal	incomplete	satisfactory	good	excellent
The structure of the report is ...				
abysmal	poor	average	good	excellent
The quality and style of the grammar used in the report is ...				
abysmal	poor	average	good	excellent
The quality, relevance and referencing of the tables and diagrams is ...				
abysmal	poor	average	good	excellent
The use of references is ...				
abysmal	poor	average	good	excellent
The way in which the discussion and conclusions of the project are presented is ...				
abysmal	poor	average	good	excellent
The number of errors in the report is ...				
excessive	numerous	acceptable	small	almost none
The definition of the project area is ...				
non-existent	vague	adequate	succinct	sharply focused
To what extent are the ideas presented in a natural and orderly way?				
not at all	sometimes	satisfactorily	mostly	always
The review of the theoretical background to the project is ...				
non-existent	vague	adequate	succinct	sharply focused
The report identifies the following proportion of the main issues ...				
none	some	about half	most	all

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>





The analysis is ...				
vacuous	feeble	adequate	good	insightful
The report explains the rationale behind the design of any artefact ...				
not at all	ineffectively	very adequately	effectively	very effectively
The report compares alternative designs and justifies choices ...				
not at all	ineffectively	very adequately	effectively	very effectively
The report suggests that the project was ...				
trivial	feeble	adequate	challenging	impossible
The evaluation of the work done was ...				
vacuous	feeble	adequate	thorough	rigorous
The relationship between theory and practice was identified ...				
not at all	weakly	adequately	strongly	everywhere

These forms are then all returned to the assessor(s), and are used to inform their validation of the numeric mark(s) awarded. ***It works better if*** all assessors are involved.

It doesn't work with particularly strong-willed individuals who will mark according to their own private criteria whatever guidance is given.

---ooOoo---

So: consider separating numeric from subjective grading, so that one type of information informs and enhances the other.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

6.9 Never make a choice without a reason

Students often consider that their projectwork exists in isolation, and don't bother to situate it in the wider world.

---ooOoo---

This bundle is concerned with ensuring that students undertake an appropriate amount of background reading.

The way it works is that students are required to deliver for assessment a justification of choices taken and decisions made against external material. They are required to evidence background reading in their project deliverables.

The most common occurrence of this is through the use of a literature survey in a Research-type project [see 1.5]. However, it can have different qualities. For instance a discussion of the merits of different notations, and a rationale for any choices made, may form a (required) part of the design documentation. This can also be a requirement if a structured analysis and design method is used, for example within the *Selection of Technical Options* stage in Structured Systems Analysis and Design Methodology (SSADM).

It works better if there is a shared understanding (between staff and between staff and students) that no work exists in isolation. This can be successfully addressed in other teaching with regard to the ethical and "professional" aspects of academic work (such as the necessity for proper attribution and the problems of plagiarism) and related to their work on a project.

It doesn't work unless the requirement to produce the evidence is acknowledged in assessment.

---ooOoo---

So: consider assessing students' understanding of the context of their project





Reflection

Reflection on experience underpins the process of successful learning and is essential to the success of education. The major problem here lies in the process being well removed from the technical content on which students fixate, and the fact that it is its own reward - allocating "marks" for reflective activity is likely only to divert attention from its true purpose.

While supervisors may be well aware of this, all too often students are not; reflection is an activity that does not carry explicit credit and it can be hard to convince students that the benefits of it will accrue, not least, ultimately, in raw grades. Since, to the inexperienced, the merits of reflection are not evident, there is sense in devising activities surrounding project conduct that will cause it to happen, perhaps covertly. It is common to see groups looking inwards, or for individual students to become over-absorbed, and coaxing them out of this is a supervisory duty that can have beneficial side-effects in changing long-term practice. Another, lesser, problem is that students may well engage in reflective activity without realising it, and hence not learning the merits of this approach to learning. This lends purpose to activities that involve the articulation of reflection.

It is probable that the project is the first truly large-scale piece of work a student has undertaken, and the adjustment to this mode of working can be difficult, or at least not obvious. Developing the skills to handle intermediate deliverables that may or may not be cumulative and to work in activities where interaction is positively beneficial is not easy or obvious. This is particularly true if all assessed work hitherto has been strictly individual, with penalties for plagiarism, and smaller scale, operating over no more than a few weeks. The transition can be significantly eased by a reflection on one's own knowledge, skills and working practices, followed by a suitable adjustment based on what is said.

The other aspect of scale is the longevity of the work. Quite unlike course-work, project work may well contribute to larger developments that have a longer and altogether more important existence. Where this is not the case, however, it is very useful to suggest to students that project deliverables can have a visible longer lifetime. If, at completion, the work is not mentally shelved it can be used as the basis of a reflective experience long after assessment and departure from the university.

Projects are conducted in a wide range of environments, some of which provide more support to encouraging reflection (and other desirable skills) than others. In particular, the active support of a university library can be a great spur to making the activity "serious" [see, for example, 7.7 *Cherish it*], but staff taking matters "seriously" at a departmental level can suffice. This requires broad agreement on what the project exercise is for (not just in written aims and objectives!), and can be encouraged by a

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

visible administrative and support structure that stands independent of, for example, course-work management.

There are many ideas that can be used in this area; a popular one is the logbook, which can, if properly implemented, have the full reflective effect of a personal diary, especially when there is a requirement to revisit it from time to time. Many formats are possible (paper, HTML, etc.), but a written journal provides discipline in expression and permits portability, especially as raw material is likely to be electronically available somewhere else. If this is used, it is a good idea, especially at the outset, to require a specific format since this will assist those inexperienced in this kind of activity; this is particularly true if students have difficulty focusing on processes. Displaying a good example from earlier years is also useful, although there is often a problem with re-use of formats and words that are perceived by students to be "successful".



Logbooks can fail if students see them merely as an overhead on the project? a distraction from the "real" work of software production, and will not work effectively without adequate motivation and reward for the student. The reflection can be little more than superficial, particularly if the log does not hold much information. There is also a danger that it becomes literally a log ?i.e., simply a history of what happened with no analysis or conclusions for the future. Without this analysis the log does not provide the full value for any reflective reports based on it. Significant events can occur in projects that no tutors are aware of; consequently the log may not include events which produce significant outcomes. It should not be expected that the log will contain everything of significance.

The general idea is applicable to group-based work as well; a team based log may suffer particularly from incompleteness due to reluctance of students to report problems associated with peers. The inverse problem may also appear, i.e., that one student's experience dominates the log [See 8.5 *Red card/yellow card* and 8.6 *Moderation using student input.*]

Projects frequently require a range or number of deliverables; the "report" and/or a software product are common, but the specifics can vary and other options are available. The possibility of phasing deliverables provides an excellent opportunity for reflection and adjustment of practice where necessary. For example, students may be required to produce a progress report, a literature survey, a draft chapter of the final report, or a proposed future schedule. Any or all of these can be used as the basis of supervisory meetings and have great value in spurring reflection since they require an examination of achievement to date. Going one step further, such submissions may be made the subject of formal assessment (preferably by more than just the supervisor)?upon return, invaluable advice can be provided on how things are going. The return of the assessors' comments, while the project is still underway, can provoke self-examination and adjustment, and provide a handle for future reflection on progress.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

The project experience is usually long, and provides many opportunities for reflection of various kinds;

- Before: At project outset, it is likely that there will be some experience on which to build. Constructing explicit reflection before the "real" work has commenced can be illustrative and good experience of the general practice.
- During: Projects may well run for some months, a major difference to coursework. There is therefore the opportunity for reflection on progress and adjustment of practice.
- After: Most of a project's life is after completion, but it is all too common for it to be put on a shelf and forgotten. There are many opportunities to use the project experience, both for the author and other students, and maximising these aids reflection on the whole process.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.1 Throw the driver under the bus

Teams/groups should be resilient to changes in personnel, but in practice students often rely on the particular skills of a small number of key individuals.

---ooOoo---

This bundle requires students to reflect explicitly on who can do what, by disturbing (or temporarily destroying) the natural or established allocation of duties.

The way it works is to set a project that requires different skills to be deployed, and at a crucial moment, remove from the group an individual with a particular set of skills. (For example, set a programming task then remove the good programmers). This determines whether someone other than the driver can keep the bus moving along the road. Once the person has been removed, ask the rest of the group to modify some of the person's work. Keep the key person occupied on some other activity. The task set while the person is removed needs to be small enough to be completed in the time available (e.g. hour, or half day). This can also be extended to several groups.

It works better if the activity requires (and the group possesses) a diverse mixture of skills ? for example, on a highly inter-disciplinary course. For reasons of economy (and surprise) it is better, where this is applied to a number of groups, they are in some form of synchrony ? for example if all the groups are conducting the same work to the same schedule. It also works better if you have a good understanding of the dynamics of each group.

A variation is to swap students with the same key skill between groups.

It doesn't work unless you monitor how the groups tackle the problem: some may subcontract the task, others may complete it effortlessly using records that the key person left behind, and some groups may fabricate the results. Monitoring is important because you need to explain to students what has happened, and get them to reflect on it.

---ooOoo---

So: ensure that groups recognise the value (and use of) process documentation, by making them rely on it.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.2 Mid-project report

If students have no formal feedback during the conduct of a project, there is no guide to keeping the show on the rails, and no spur to commencing or conducting the reflective process.

---ooOoo---

This bundle requires reflective activity to begin well before project completion.

The way it works is to require students to deliver partway through the project an interim report. This should be significant but not overwhelming, and should be submitted to a strict deadline and assessed, preferably by more than just the supervisor. It will provide an early-warning of impending problems, both among students and potential disagreements between assessors. Upon return to the student, invaluable advice can be provided on how things are going.

This practice is of great value in spurring reflection, since such a report requires an examination of achievement to date; furthermore, the return of the assessors' comments, while the project is still underway, can provoke self-examination and adjustment, and provide a handle for future reflection on progress.

It works better if conducted a little before halfway through the project, with a swift turn-round. It also helps if the feedback contains comment on the perceived level of project difficulty (and ultimate maximum grade attainable)

It doesn't work unless you have sufficient resource, since the deliverable needs co-ordinating across the cohort, and staff time needs to be devoted to assessment. It is important that this is done briskly or the benefits are diluted.

---ooOoo---

So: to require reflection to commence, introduce an interim, reflection-based, deliverable; feed it back with constructive comment and guidance.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.3 Co-ordinated supervision

The most accessible, and sometimes best, advice can come from the peer group. Student working practices do not always encourage this, while project work should engender the practice of asking for and providing support.



---ooOoo---

This bundle requires students to account for their progress and discuss problems, with each other.

The way it works is by the supervisor requiring groups of individual project students to meet in a structured setting, usually meaning a minuted meeting of fixed duration?an hour normally suffices.

Working partnerships can grow spontaneously, and mature problem solving mechanisms can be developed. The formal presentation of progress requires student reflection on their achievement and plans and usually provokes conversation and debate on possibilities that are often outside the supervisor's experience.

If the group meetings are formal (minuted) and of manageable size - 5 students can meet a supervisor and exchange business within an hour. The peer pressure to attend with something positive to say is an aid that is quite absent in individual supervisions. There can also be a significant time saving over a sequence of individual supervisions, but it is important not to let the students see this as a motivation for the practice.

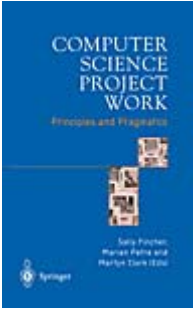
A variant is to rotate the tasks of chair and secretary around the students, thereby giving them experience of these organisational roles. Another variation is to alternate group meetings with individual supervisions; this provides reassurance to those who might feel the need of specialist input.

This may not work if the group of students are studying projects in disparate areas, since they may not have interest or knowledge in the work of some others. In fact, this idea has been seen to work well in such scenarios but it is necessary for the supervisor to ensure the organisational benefits are drawn out, and that no "lame ducks" develop. It may also, therefore, not work if the supervisor is wedded to the idea of 1-1 supervision.

---ooOoo---

So: devise ways of encouraging "vicarious learning" by requiring students to explain their project triumphs and obstacles to each other.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



7.4 Project log

The raw skills of reflection are often absent or under-rehearsed in students.

---ooOoo---

This bundle initiates reflective activity at project outset, and maintains it throughout.

The way it works is to make a diary (or log) a deliverable of the project. Use the content of the log of the immediately preceding week(s) as the agenda paper for supervisory meetings, thus requiring the student to reflect in conversation on what is being done. Furthermore, when the final report is written, the diary over a period of time should illustrate the scale and nature of achievement, and provides hard data for reflection on what has (and has not) been conducted.

The same idea can work just as well in group projects. A decision needs to be taken on whether there is one log for the group, or a number of individual logs, or a hybrid of the two approaches.

The scale of credit allocated to the log need not be great, since most credit will be won for achievements made whose progress it records. The credit does need to be sufficient to ensure the exercise is taken seriously.

It works better when there are clear intermediate milestones of the project that provoke explicit reflective recording.

It doesn't work if students see the log as an overhead on the project

---ooOoo---

So: catalyse reflection by requiring it to be recorded.

7.5 Sooner rather than later

Although reflection can often lead to an improvement in students' performance during the project, such opportunities will be lost if students view reflection as a process which only takes place after project work is completed.

---ooOoo---

This bundle provides a structured mechanism within which students must exercise reflection through the conduct of the project.

The way it works is to have students use Watts Humphrey's Personal Software Process (Humphreys, 1997) to record the effort they have expended on individual tasks within a project, to use these records to assist them reflect on their work, to implement changes in the way they approach their work, and to track the effects of the changes.

This bundle fits within a project in which students are required to design, build and test a number of software modules, or similar deliverables. If used repeatedly, more benefits can accrue: assessors feel that students are working steadily producing real quantitative results and, over time, students development strategies change.

It works better if the student is not penalised for attempting process improvement which does not have a beneficial outcome.

It doesn't work when unless the project requires students to undertake a task or group of tasks on a number of occasions. It does not work if students leave the work until the last moment, and it requires a standardised set of documents. It doesn't work if students are able to exploit opportunities to fabricate results.

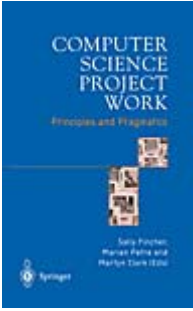
Reference Humphreys, W. S. (1997) Introduction to the personal software process, Addison-Wesley, Reading, Mass

---ooOoo---

So: employ mechanisms which require students to shorten the cycle of reflection



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



7.6 “Follow that plan”

Work schedules stretching over weeks or months are new to most students, who have little experience of estimating or measuring how long particular tasks may take.

---ooOoo---

This bundle requires students to produce a time plan, to monitor their progress against it, and to amend it as necessary, noting the changes as they go.

The way it works is to make the production of a time plan the first project activity. This is then used on a regular basis in meetings to monitor progress. It is probable that modifications to the plan will become necessary as a result of project developments, or poor estimates; this provokes immediate reflection on how and why things need to be changed. At the end of the project, students are required to evaluate their performance on the project in respect of the project planning and time management skills.

This idea can be used in any project situation, individual or team-based, where the work can be split into separate tasks.

A side effect is assisting time management skills, and improved understanding of the estimating of time taken to complete a task.

It works better if students have experience of following, or are strongly encouraged to follow, a clear, phased approach to project development.

---ooOoo---

So: find ways to ensure the time consumed by project tasks is estimated, measured, and commented upon.

7.7 Cherish it

At project completion, many of the benefits of project experience can be lost if students think the exercise is "all over".

---ooOoo---

This bundle encourages students to see project products as having a lifetime beyond assessment, and their own university careers; this encourages greater pride in their work and more reflection on its production.

The way it works is to require project reports to be submitted in duplicate. After assessment, one is returned to the student and the other is filed somewhere visible for future reference. Electronic access could also work, and would have the merit of immediate international access. When students see their own work on (permanent or semi-permanent) public display, they are encouraged to take pride in it and may well re-visit it from time to time. Such visits nearly always result in increasingly mature reflection on what they did, often with the benefit of intervening experience.

Over time, success can be judged by students examining the work of others and remarking "I could do better than that", or similar, or if they draw attention to the hazards and benefits of public availability during production. A more immediate test of success is an accumulation of reports from earlier years.

It works better if students are required to read one or more reports from earlier years from the accessible pool, since foreknowledge that the project deliverables will be filed publicly can also affect the sense of pride students take in project conduct.

It doesn't work if for some reason a project is of low quality, in which case "image" problems in its public availability may be expressed. It is possible that easy access to reports and grades may provoke comparisons leading to appeals (justified and unjustified), or that those taking especial pride may wish to update/edit their reports before public filing. If paper copies are filed for public access, significant filing space is a necessity, and the material often (usually) ages after 5 or so years.

See also: 9.2 *Well they managed*

---ooOoo---

So: ensure something outlives the project that students can point to and cherish



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.8 “I thought that ... ”

Reflection on the project process should include staff. Often this does not occur even when students do participate.

---ooOoo---

This bundle encourages students to reflect formally on the project process, and provides material to assist staff and project managers in reviewing and improving it.

The way it works is to encourage (or require) students, on completion, to complete a survey in questionnaire form of their perception of project process and outcomes. Precise questions asked will depend upon local feelings about shortcomings or problems in affairs, but standard guidelines in the preparation of questionnaires (Oppenheim, 1992) obviously apply. The act of completing the exercise is of immediate benefit to the students, and the outcomes usually point to adjustments that departments might make to improve. The outcomes can also be of value as pre-publicity to students

It works better if the largest possible percentage of the cohort can be surveyed; one approach is to make the return of a private copy of the project report contingent on this action having been taken. It also assists if students operate in a culture of responsible feedback in which coercion is not necessary to encourage them to state frank and responsible views.

It doesn't work if too much credence is given to student opinion, which is often an unreliable source of quality information, and it may not work if the survey is conducted prior to assessment, when frank opinions may not be forthcoming. There may be a problem in extracting opinions after assessment when students are under-motivated to engage in this kind of exercise.

Reference Oppenheim, A. N. (1992) Questionnaire Design, Interviewing and Attitude Measurement, Pinter, London

---ooOoo---

So: obtain the widest inputs you can to surveys of project process and outcomes. Use them to improve and inform, and to initiate reflective thought.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.9 Last year's punters

The project provides a wealth of experience, often intangible, that might not be recognised. Simultaneously, this experience is of potential value to junior students and may go untapped.

---ooOoo---

This bundle recognises that a lot of the process of project conduct is learned very well during its course, but this may go without proper recognition by the student. It is also the sort of experience that is very hard to "teach", but may well be communicated effectively by a peer.

The way it works is by realising that both those problems are soluble by facilitating interaction between students at different levels. One way of doing this is to make a presentation or poster a project deliverable, and getting the junior students to take part in assessment, although actual interaction between the groups helps more. Getting senior students to talk to groups (large or small) of junior students about "how it was for me" is another variant. This may be done as part of a formal lecture, or to *ad hoc* groups in, for example, tutorials. If the group is small enough, very fruitful conversation can ensue. Of necessity, the senior students must have thought about their experience, and thus reflection is stimulated.

It is necessary to ensure that the presenting students have actually conducted the reflection necessary to inspire such interaction.

A side effect is rehearsal of presentation techniques.

You'll know this is working if junior students make remarks such as "That chap warned me that this might happen", or if senior students remark on how much they learned outside the technical necessities.

It doesn't work if there are problems in scheduling such interactions, especially if senior students are at the end of their university careers soon after project submission. It is also possible that they will communicate things that contradict what supervisors might wish.

A variant is to use newly recruited PhD students who will in all probability have recent project experience. Possible problems with this are that their experience may not map directly onto local conditions, and that they are self-selecting in being at the highly academic end of their cohort, and may not communicate what is required by the broader mass.

Another variant was given by one of the anonymous reviewers of this book: "My particular variant of reading *Last year's punters* is to introduce an open day, when



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

finishing students can demonstrate their products, both to staff and students from the previous year. This will introduce an aspect of continuity and inter-year contact."

---ooOoo---

So: find ways of getting last year's students to talk to this year's, to the benefit of both.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

7.10 “If I had my time again”

Students rarely re-examine their project progress and ask how they might have done it differently with improved (or different) outcome.

---ooOoo---

This bundle makes explicit reflective activity a deliverable by asking students to note opinion on the whole project process.

The way it works is to require a section of the project report to address the question "How might the work have been conducted differently?". Some element of guidance is necessary when encouraging answers to this, to prevent responses being at a superficial or purely technical level.

It is possible to ask for other questions be addressed as well - "What have you learned?" is a good example.

It works better if there has been genuine reflection on the whole project process, perhaps, but not necessarily, conducted within a supervisory session. It is even better if this is conducted in company with other students of the same cohort, since this encourages them to see differences and similarities in their individual experiences.

It doesn't work unless the reflective practice is genuine. This can easily be undermined of, for example, the responses are seen as part of assessment, in which case formulaic phrases from earlier years that are seen as "successful" may well be delivered instead of true opinion.

---ooOoo---

So: devise ways to make students answer the question "What if I had done it differently?"



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



Team/Group projects

It is a truth universally acknowledged by employers and professional bodies alike that an element of group work in a practical subject like Computer Science is A Good Thing. Yet group work sometimes sits uneasily in an environment where individual achievement is prized and collaborative activity (of the variety labelled "academic misconduct") is deplored.

"Group work" is the name given to a variety of projects in which groups of students collaborate to achieve a single goal. It manifests itself in laboratory work (often involving pairs of students working on small laboratory-based exercises), in so-called team projects (involving a large cohort of students divided into groups of at least three engaged in collaborative activity, often the construction of some artefact), and in places where groups of students collaborate to work on an individual project of a scale that they could not have attempted individually (often, a final-year or postgraduate activity).

Group work is valued within degree programmes, not only for its intrinsic learning goals, but also because it is perceived to inculcate in the participants an awareness of the needs, difficulties, opportunities and complexities of an activity essential to the professional engineer in the computing discipline. Conversely, although group work often has a large profile within a degree programme, the marks that it attracts may be a small proportion of a student's final classification, so students and staff have to balance effort against the potential reward.

You can't avoid process

As soon as group or team working is required in a project, then working with others?the technical and personal collaboration necessary to build a joint product?cannot be avoided. This means that students will expect support in this way of working.

Supervisors (and institutions) vary in their approach to this requirement, from the very prescriptive ways to manage process provided by continuous quality process documentation to the laissez-faire "this is what group working is, you must expect problems, just get on and do it" [see also 5.1 *Characterising Supervisor Input*]. Students may also be empowered with mechanisms to address internal group problems which do not require supervisor intervention [for example, see 8.5 *Red Card/Yellow Card*].

Whatever approach (in whatever degree) is taken, the simple fact of the way in which the students are working (in groups) adds a dimension which is quite absent from

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

typical forms of learning in the rest of the curriculum ? that is on coursework or on individual projects.

Devising activities suitable for groups from within Academia

This involves devising activities that are suitable for groups of students to complete in the time available, while still achieving the educational objectives. The work typically involves exposing students to as much of the product lifecycle as possible, yet remaining small enough to allow students both to encounter challenges and have time to reflect on their achievements (or otherwise) at the end of the activity [see section seven *Reflection*, especially 7.7 *Cherish it*]. The activity must sub-divide to enable a division of the task between the members of a group. Where a cohort is large, there will typically be many groups attempting the same task, which must therefore be capable of solution in a variety of ways to avoid excessive duplication between groups.

... from external sources

A desire for realism may encourage the supervisor to consider introducing some element of "real world" input. This can be a mixed blessing: the obvious advantage of exposing students to a problem that they might have tackled had they been in employment is to be welcomed, but input from companies (say) is necessarily circumscribed by real commercial constraints. This might mean that the scale or scope of the project proposed, or the outcomes expected (i.e. a robust commercial quality product) are unrealistic [See 4.2 *"I'd like to do that"*]. Other problems can occur in the student-industry interaction. For example, a response to a student question from an external "client" or "manager" is often not as prompt nor as detailed as a local supervisor might give. All of these problems can be useful educational experiences, of course, assuming that there is an opportunity for appropriate reflection and debriefing. In practice, real world input is often most effective when moderated by someone in close contact with the students tackling the project [see 3.3 *Creating a real company*].

Achieving balanced groups

The key problem here is to attempt to allocate students to groups so that each group has an equal chance of doing well. Those who have read Belbin's (Belbin, 1981) work on the structure of teams will appreciate that the aim is to form teams that have a good balance of skills; this is not necessarily the same as a good balance of academic ability. Students easily perceive that allocations are unfair, typically because they perceive one team as having more academically able students than another. Staff, however, perceive that this can be unfair for different reasons ? a team with the most academically able students in it may be one with too many strong characters to form an effective and harmonious team. It has been shown (Thorn, 1998) that when



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics* , Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



allowed to self-select, students do not work to maximise the skill set of the group for the task in hand, but instead choose to work with their friends.

Getting started

Once the allocations are done, all groups need an initialisation strategy to get them going. The issue here is whether to let the students simply get on with the task set (which may be too much to expect of less mature students), and hope either that a natural leader will emerge to take control, or appoint someone with the specific task of leading the group. Another, less intrusive method is to require that the team have named roles, but let them decide (by whatever mechanism they choose) who is to take on which role. If tasks are specifically allocated, they may rotate between members of the team or be negotiated with the supervisor; an election or job application process may also take place. Whatever your choice, it is helpful to students if you tell them which initialisation strategy you are adopting. 3.2 *Roles in groups* has more details.

Consistency in supervision

The issues surrounding the supervision of group projects can be clustered around the role of staff, the activities of students, and a small number of special circumstances. Where a large cohort is engaged in a group project it is likely that several staff will need to be involved in supervising the cohort. Where different staff have different skills, experiences and approaches, it can be difficult to get all the staff to give advice in a consistent way. Recognising and using those different skills, for example by allocating staff to a functional role rather than to a group of students, can mitigate this. [See also: 2.12, *Staff deployment*, 4.1 *Me and my shadow* and 5.1 *Characterising Supervisor Input*].

Whatever role is allocated to staff, staff need to check that every student within a group is making a contribution, and that the group as a whole has the opportunity to reflect on the process as it proceeds. Occasionally a student will become isolated from the rest of the group. Sometimes this is because they think they can complete the task on their own (which may be true, but that's not the point of the exercise), sometimes this is because their personality or culture makes it difficult for them to interact effectively with their peers, or sometimes it happens that other team members become unavailable (for example, through illness) [See also section five *Stuff Happens*]. Supervisors need to be aware of these problems and offer appropriate encouragement; they should also make sure that groups are not so small that they become individuals [Although see : 9.1 *Going solo*].

Individual credit for group effort

Of all the issues surrounding group work, that of assessment is perhaps the most problematical because it is the issue on which students, staff, institutions, professional bodies and external examiners alike have the strongest views.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This “Field Edition” of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

It must be a prerequisite that it is acceptable that marks awarded for group activity can be counted towards the degree classification of individual students. The extent to which they count varies: 5% would be unexceptional, more than 20% may require considerable discussion. It is common for students to engage in group activity, but nevertheless be assessed, at least in part, by an individual assessment reflecting on the experience [see also 6.1 *Use Peer Assessment*].



Where identical marks are given to individuals for the work of a group, it is often for the construction of an artefact, but this raises the issue of whether all students made an equal contribution to the activity. This can be addressed by moderating the marks awarded for the group activity by student input, allowing a group of students to agree that individuals made greater or lesser contributions to the activity. Getting the students to agree, or where they cannot agree, having a clear method of resolution, is key [See also 2.9(ii) *Group assessment*]. Where a group project has involved industrial input, an industrial contribution to the assessment may be appropriate. Here, a shared understanding between the academic and industrial views is essential. Industrial input in the form of a prize may mitigate potential differences between academic and industrial views [See also 2.8 *Motivation*].

Did they get the point?

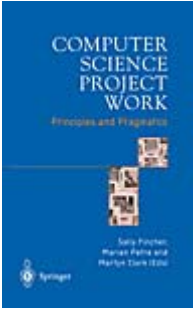
At the end of the project, the participants should have an awareness of the needs, difficulties, opportunities and complexities of working as a group. If you are the supervisor, you need to know whether this has happened. You also need to know whether the activity was too easy (so that students have not experienced enough of the problems), too challenging (so that students have been over-occupied with too limited a range of activities) or just right.

If the project was just right, students will feel that they have been challenged to face problems, but have been able to overcome them, and that they gained understanding and useful experience. You will know all this by having been involved in the project, by soliciting feedback from students and colleagues, and engaging in your own reflective activity. More negatively, some of your students may have hated the group work experience, so reflection, and getting them to realise that they may have learned something from the experience, is important. At its most positive, your students will obtain jobs by being able to articulate their experiences of group work in an interview, and this seems to be important to students (and potential employers) at the present time [See also 7.10 *"If I had my time again"*].

References

- Belbin, R. M. (1981) *Management teams: why they succeed or fail*, Heinemann.
Thorn, K. (1998) In *Projects in the Computing Curriculum* (Eds, Holcombe, M., Stratton, A., Fincher, S. and Griffiths, G.) Springer-Verlag, London, pp. 217-224.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



8.1 Managing staff input

Group projects are often of such a scale that several members of staff need to be involved in supporting students doing the project. Members of staff need to give consistent advice (in terms of content and amount) to all teams.

---ooOoo---

This bundle mitigates potential differences in the input that different staff can provide to different groups of students.

The way it works is that, rather than allocate members of staff to individual groups, they are allocated by functional role (for example, one to deal with requirements specification issues, another with implementation and another with testing; or in a multi-disciplinary project, one to act as finance director, another as technical director, and so on). Student interaction with staff is by e-mail only, and by their role, not as individuals, making it easier for individual members of staff to give consistent advice across a single functional area.

It works better if you have available staff with appropriate functional skills, but it can also help use staff who lack the breadth of skills to advise a group throughout the complete project lifecycle.

It doesn't work if students would really benefit from the physical presence of a member of staff in the laboratory (for example, because very detailed instructions need to be given), though staff can be briefed to tell students to ask detailed questions via the e-mail system.

See also: 5.4 *Supervisor's Eyes and Ears* and 5.3 *The help they've had along the way*

---ooOoo---

So: give careful consideration to *which* staff are to be deployed in managing group projects, and *how* that deployment might be most effective.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

8.2 Fair allocation

It is important to allocate students to groups fairly.

---ooOoo---

This bundle allocates students to groups so that no group is perceived to be stronger or "better" than any other.

The way it works is that the supervisor compiles a list of the students in the cohort ranked by ability (for example, by the previous year's examination marks). If the cohort needs to be divided into G groups, numbered 0 to $G-1$, the student ranked R is put in group $(R \bmod G)$. If this initial division produces a group that the supervisor knows is unsatisfactory (for example, it comprises a group of close friends, or has unfortunate gender divisions) some tweaking by hand may be beneficial to achieve the final allocation.

Alternatively, you can simply rank the students in alphabetical order and divide as above.

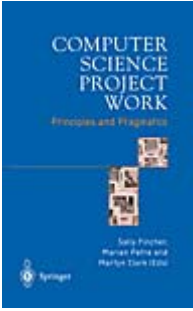
It works better if you have a large cohort of students; it is also a very cheap mechanism to implement and perceived to be no less fair than any other random allocation.

It doesn't work unless you assume (perhaps erroneously) that examination marks are a guide to how well individuals will perform in a team context.

---ooOoo---

So: use an allocation mechanism that disadvantages neither an individual student nor a student group.





8.3 Maximal allocation

In group projects it is necessary to divide a cohort of students into groups. Students and staff would like each group to be equally able to complete the group task. It is important to maximise each group's chances of doing well.

---ooOoo---

This bundle helps to identify individual characteristics that might be combined to make successful teams.

The way it works is that students complete a questionnaire ? due to Belbin (Belbin, 1981)? that will form a profile of their potential contribution to the team across eight broad areas: implementer (makes decisions or plans happen in a sensible and practical way), coordinator (makes sure everyone is clear about what is to be done), shaper (driven by urge to get things done), plant (source of new and unusual ideas, suggestions and plans), resource investigator (makes friends easily and has masses of contacts), monitor-evaluator (steady person who thinks things through), team worker (makes sure everyone works together well) and finisher (meets deadlines at all costs).

Use the scores for individual students to allocate students to groups so that each group contains one coordinator, one strong plant or shaper, one finisher, one team worker and one monitor-evaluator.

It works better if no group contains two strong plants without a very strong coordinator, and also that the groups are roughly comparable in overall technical ability. **It doesn't work unless** you read one of the texts that include a copy of the Belbin questionnaire, and an explanation of the scoring system. It also requires a fairly large cohort of students and is more time-consuming to administer than some other allocation mechanisms. It is useful, however, for students to use the questionnaire to reflect on their own traits. Sometimes the distribution of Belbin categories is very uneven.

Variation: Alan Jones from the University of Teesside has used this technique in a Master's level course. He describes it in: Experiences of Profile-Based Group Composition in the journal *Computer Science Education* 1999, Vol. 9, No. 3, pp. 242-255 **References** Belbin, R. M. (1981) Management teams: why they succeed or fail, Heinemann.

---ooOoo---

So: invest a little time in reading about Belbin's questionnaire and use it to build good teams.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

8.4 Battle-scarred veteran

When students undertake their first group project, they often have no idea of where to start or how to proceed.

---ooOoo---

This bundle provides the group with experience by introducing, as a leader, a student from a later year who has undertaken (sometimes several) group projects in similar contexts.

The way it works is that the earlier-year students are allocated to groups by the normal process. Each group is then allocated a later-year student to act as leader. (If there are more students in the later year than there are groups in the earlier year, another exercise will have to be devised for those who have no group to manage. Also, a selection process must be established ? for example writing an application: "Why I want to do this, and why I would be good at it"). The later-year students provide what input they think the project needs.

It works better if the project is reasonably short (5-6 weeks). It works better over time, as the managing students will have experienced "being managed" earlier in their programme (and are often eager to participate). It works better if The Management is also part of a credit-bearing course (for which the assessment might be a log of the project, and/or a reflective piece). If such a system is adopted, a collateral benefit of having the later-year management logbooks is that it makes staff assessment the project itself much easier.

It doesn't work unless the two cohorts are available at the same time of the year.

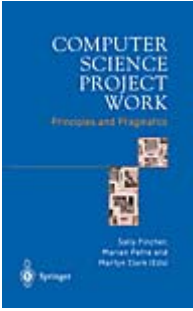
See also: 7.9 Last Year's Punters

---ooOoo---

So: provide support for initial group work from a perspective with which the students can readily identify



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



8.5 Red card / yellow card

Students and staff alike are reluctant to reward group members who do not contribute, although some groups seem perfectly happy to "carry" a hitch-hiker. In either case, it is impossible for staff to know precisely how much work each team member did: only the students involved know this.

---ooOoo---

This bundle gives students some control over the behaviour of members of their project group and allows their non-performance to be factored into assessment.

The way it works is that students are allowed to issue others in their project group with yellow, and *in extremis*, red cards. A yellow card is "shown" to a student who is deficient in effort or attitude or in other ways not making a full contribution to the group and is then lodged with the project supervisor. Being "shown a yellow card" results in a known penalty being applied to the student (for example a fixed number of marks lost), though a yellow card may be cancelled by increased effort, or at a boundary between phases of the project, or after a set time. A student who attracts the maximum number of yellow cards can be "shown a red card", which excludes the student from the rest of the project and sets the mark awarded to zero. There is no recovery from a red card.

These cards are public documents and their issue is a formal process. It must be ratified by a majority of the students in the group, and the final step of the procedure (actually issuing the card) can only be undertaken by the supervisor. A collateral benefit of this process is that the number of yellow cards received by a student across a programme can be used as a monitoring tool.

It works better if staff set the parameters of control (the penalty, the number of yellow cards that can be carried)

It doesn't work if the system leads to the frivolous use of penalties. It doesn't work unless day-to-day management of the resource/role allocation is in the hands of the group themselves.

---ooOoo---

So: find a mechanism which devolves some control over the performance of group members to the groups themselves.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

8.6 Moderation using student input

Students' awareness of their own group processes can be poor (or absent). It is difficult for staff to require (or encourage) such awareness without giving students both tools, and the motivation to use them.



---ooOoo---

This bundle allows students to give feedback on the contribution made by each team member (including themselves) under headings related to both the technical and managerial aspects of their work.

The way it works is that Peer and Self Evaluation forms are issued to team members after the submission of each deliverable. Each student completes (anonymously) a Form for each member of their team and for themselves. For each student, the response to each of the scaled questions is scored (-3, 0, +3) and an average taken. Within each team, individual average scores are compared with the team average. The supervisor moderates the individual mark for the assessment (by a maximum of $\pm 5\%$ from the team mark) for students with a large deviation, taking additional account of free-form comments on the Form.

Peer-and-Self Evaluation Form

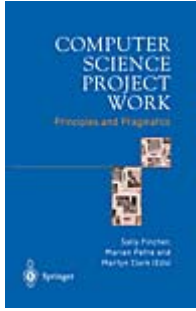
You should complete this **Peer & Self Evaluation** form for **each member of your team and yourself** and hand them in to the Departmental Office in the normal way.

Name of Student:

Please indicate for the team member their management characteristics:

time management	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	highly organised			unreliable			
responsiveness to others	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	respects views			domineering			
coping with stress	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	always calm			panics easily			
decision making	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	decisive			unable to commit			
co-operation	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



	always co-operates					goes own way
self confidence	<input type="text"/>					
	able to take criticism					can't take criticism
leadership	<input type="text"/>					
	takes initiative					follows others
problem analysis	<input type="text"/>					
	incisive					woolly
project management	<input type="text"/>					
	best practice					activity lacks co-ordination
project evaluation	<input type="text"/>					
	systematic and objective					casual and subjective

Comments :

Please indicate for the team member their technical contribution:

Task Analysis	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
Conceptual Design	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
VDM	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
Manager's Meetings	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
Team Meetings	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
Low Level Design	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
	well above average		average			well below average
Coding	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



	well above average		average			well below average
Testing						
Documentation						
Demonstration						

Comments :

It works better if students have training in how to assess, and are practised at reflecting on process issues.

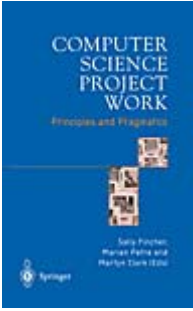
It doesn't work unless students are producing at least one substantial, well documented group-produced deliverable.

See also: 3.5 *Emphasis on Personal Transferable Skills*

---ooOoo---

So: use Peer-and-Self Evaluation Forms to moderate group marks according to individual contribution.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



8.7 Quick off the mark

When students have been allocated to groups they often spend the early part of the project not knowing quite what to do. They will achieve more if the roles they are expected to take are made explicit, and they are encouraged to take a role to which they feel suited.

---ooOoo---

This bundle helps groups get started by encouraging each member to apply for a specific role in a group.

The way it works is that a number of roles are described by means of job descriptions. The roles can be chosen from managerial or technical specialisms: manager/coordinator/leader, requirements analyser, designer, implementer, tester, interfacier, integrator, researcher, secretary, librarian, resource controller. Job descriptions are publicised and students invited to identify a small number of roles to which they feel themselves suited. The students are allocated to groups by particular jobs within the group.

It works better if there is a good balance between what students feel themselves suitable for and the roles that need to be filled.

It doesn't work if the students have to cover a broad range of activities within the group rather than concentrating on one particular role.

---ooOoo---

So: find ways of making group roles explicit

Motivation: The Achilles Heel of Learning



There are only a limited number of things staff can do ? interventions they can make or advice they can give ? which will affect motivation. Where there is no, or low, motivation in a student there is little or no learning. Paradoxically, where student motivation is high, students can overcome most difficulties, many deficiencies (in the teaching process) and some disasters.

The best projects undoubtedly result from real interest and enthusiasm on the part of the student. Traditionally it might be expected that intrinsic motivation is not a problem for students in higher education, and, indeed, for many this is still the case. For them, projectwork is frequently very enabling ? where previously their performance may have considerably exceeded the expected standard, with projects they can achieve as highly as they wish (Roberts, 2000). However, it is recognised increasingly that, as the number of students in higher education grows and as the range of abilities broadens, project supervisors must be more proactive in helping students to find the motivation to perform to the best of their ability.

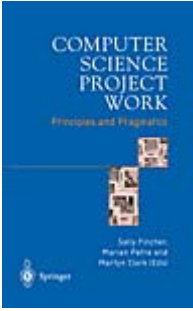
Although individuals are different, students' motivation is often related to feeling that what they are doing is valued by others and feeling a sense of partnership with the person, or people, for whom the work is being done. So, although extrinsic motivators (such as awarding a prize for the best project and other types of rewards [see: 2.8 *Motivation*] may have an impact, they are likely to focus students' efforts on earning the rewards rather than on learning and achievement. Rather, the best work is likely to result from students being able to motivate themselves intrinsically. That is, each individual should find personal satisfaction in achieving whatever it is they set out to do.

Motivation may be addressed at several points in the cycle of learning. Keller (Keller, 1983) identifies four particular points. Students' motivation can be affected by their *Interest* in the work, their perception of its *Relevance*, their *Expectation* that they will succeed and their *Satisfaction* in their achievement.

It is important to distinguish un-motivation, which may be associated with remarks such as "*What's the point of doing a project?*" (Interest and Relevance) from demotivation, which may result from difficulties experienced while undertaking the current project, or may be the residual effect of prior experience of project work (Expectation and Satisfaction).

Un-motivation may be associated with the degree programme as a whole rather than just the project component and this can be difficult to overcome. However, projects can represent an opportunity for individuals to exercise more control over their work,

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



particularly in terms of the scheduling and location of work and sometimes even in terms of choice of subject matter and methods to be used as well. Projects can, therefore, be a real opportunity to help individuals get motivated.

Demotivation is a likely, perhaps inevitable, consequence of failure to match the requirements of the project to the abilities and skills of the student: if the project is either too hard or too easy motivation will be lost quickly. Similarly, it is important that students understand at an early stage in the project the nature and range of the demands that they will face. Many students will become demotivated if the demands of the project turn out to be at odds with their expectations.

Giving students more control means that staff have less control

In attempting to create an environment which helps to stimulate intrinsic motivation a key issue is that *giving students more control over their work means that staff have less control*. For example, involving students in the choice of project topic [see 2.1 *allocation of topics to students* and 4.2 *I'd like to do that*] may help their motivation but it may also lead to staff working outside their areas of expertise. This may have implications for both the quality of supervision and the willingness of staff to take on supervisory responsibilities.

Similarly, it may be construed as motivating to allow students to determine the schedule for handing-in interim deliverables [see also 2.9(iii) *Basis of assessment: deliverables*]. This would allow students to take ownership by deciding when to work on the different aspects of the project requirements?but it necessarily means that staff work around the schedules devised by individual students. This will make it difficult for staff to plan their own work schedule and may bring project supervision into conflict with other aspects of staff workloads.

Clearly, it is necessary to balance staff and student control but achieving the optimal balance is problematic.

De-motivation involves contingency action

Un-motivated students are likely to have made themselves known in other parts of the course but de-motivation can be related to some aspect of the task in hand. Students become de-motivated for different reasons, and often the reasons relate to the individual's perception, rather than the reality, of how the project is going. Measures to cope with de-motivation must address the cause and this means that supervisors must be adept in identifying the real reasons for de-motivation (these may not be the most obvious or even those given by the student) and then tailoring strategy and tactics to the needs of the individual.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

Indeed, used indiscriminately, measures to improve motivation may do more harm than good. For every student motivated by competition, for example, there may be one (or more) students who are turned-off completely (Lepper, 1988). Publishing student work [see 7.7 *Cherish it*] may spur some students to ensure that it is of high quality, but for others this is an unnecessary extra pressure. There will always be some students who just do not respond to the invitation to negotiate aspects of their project.



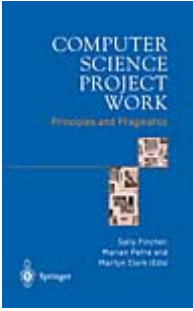
One size does not fit all

Helping students to motivate themselves is very difficult because there is no panacea; one size does not fit all, and matching ideas and approaches to individual students is a key skill for teaching staff generally and for project supervisors in particular. Further, in attempting to stimulate intrinsic motivation teaching staff can do no more than serve as a catalyst: individual students eventually must take responsibility for their own motivation. Each of the bundles in this section, therefore, comes with a "health warning": it will not work for everyone.

References

- Keller, J. M. (1983) In *Instructional-Design Theories and Models* (Ed, Reigeluth, C. M.) Lawrence Erlbaum, Hillsdale, New Jersey.
- Lepper, M. R. (1988) *Cognition and Instruction*, 5, 289-309.
- Roberts, E. (2000) In *31st SIGCSE Technical Symposium on Computer Science Education* ACM, Austin, Texas, pp. 295-299.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>



9.1 Going solo

Some students lose motivation when they find it impossible to feel ownership in a group project. In extreme cases this can mean that an individual achieves nothing and may, as a result, be in danger of failing a course which they would otherwise complete successfully

---ooOoo---

This bundle offers a contingency solution for dealing with extreme cases of group work related motivation problems.

How it works is you supervise the groups in the usual way but pay particular attention to students' reactions to being required to work in groups. Identify individuals who would be motivated to do the work but for their extreme reaction to group work. (They may manifest themselves by extreme dis-engagement and non-attendance, or perhaps by trying to "take over" and do all the work themselves, or in some other way). If undertaking this particular task in a group is not a specific requirement of the assessment criteria for the module, allow the individual in question to drop out of the group and pursue an individual project.

The individual then pursues a solo project based on the original group project but 'cut down' to be manageable by one person in the time available. The group continues as before but with a negotiated reduction in the project requirements to take account of the group being smaller.

Although it is an extreme measure, this may allow a greatly de-motivated student to achieve at least something. This may also have a beneficial effect for the group from which they drop out.

This works better if the majority of the cohort are motivated to work in groups since, once it is apparent that going solo is an option, students who to this point were working effectively in groups may want to work alone.

It doesn't work if prohibited by institutional requirements (particularly assessment criteria) and it will not work unless there is sufficient time for an individual project that satisfies institutional criteria to be instigated and pursued. Also, it must be possible for the group to continue unaffected.

---ooOoo---

So: be sensitive to how choice of teaching methods and assessment instruments can affect students' results

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

9.2 Well, *they* managed

In projects which are substantial pieces of work, students find it difficult to judge accurately the scale and scope of what is required. They may lose motivation because they cannot visualise the outcome or, if they can, find the process and the required deliverables intimidating.



---ooOoo---

This bundle offers a way to show students that other students have managed to achieve something worthwhile and that they can too, affecting confidence and motivating performance.

How it works is project reports submitted by previous cohorts are placed in the library. If you judge that a student you are supervising has motivation problems arising from an inability to visualise the outcome or by being overawed take them to the library and show them where the previous cohort's reports are kept.

Get them to take a report to read, and tell them that it will be discussed it at the next supervision meeting. At the next supervision meeting get them to talk about the project they've read, focusing the conversation on the deliverables and the processes required to achieve them. Wherever possible get the student to work out how things were done rather than telling them. Get the student to "compare and contrast" the completed project with the challenges they will face in their own project. Students get a much clearer idea of what is expected of them and find it much easier to see themselves completing the project successfully.

It works better if the ability and diligence of the current and previous student are reasonably closely matched as this helps to give the student a realistic view of what they can expect to achieve. It works better if grades of earlier projects are also available so that their lack of confidence can be attributed to real or imagined deficiencies. ***It doesn't work if*** reports generated by previous cohorts are not available and it will not work unless the current student perceives the report to be relevant to their work. Also, it will not work if staff merely deliver a lecture on how the previous student went about the project; the student should work out as much as possible for themselves.

See also: 7.7 *Cherish it*, and 7.9 *Last Year's Punters*

---ooOoo---

So: use deliverables from previous years to help students get an idea of what they have to do.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

9.3 You've done it before

A lack of motivation can be due to an individual's perception that the project is beyond their ability. This can be true even if the student has successful prior experience of project work.

---ooOoo---

This bundle aims to use the student's previous experience as a means of showing them that they can survive and thrive in project work, building on their satisfaction in previous performance to affect their expectation of future success.

How it works is if you judge that an individual's motivation problems stem from a feeling that they just cannot do it, use a supervision meeting to talk about the student's prior experience of project work.

In one-to-one conversation get them to identify other projects they have undertaken either in a group or individually. Ask them to think about what went well and what was less successful, ask about the problems encountered and how they were overcome. Get the student to make connections between prior experience and the current project.

The idea is to leave the student feeling that their previous experience is relevant to what they're doing in the current project and that they *can* do the current work.

It works better if the previous project was reasonably successful and it helps for the relevant member of staff to have detailed information about the student's previous project work.

It doesn't work if the student has no such prior experience. Also, if the previous experience of project work was completely disastrous it may not be motivating to revisit it.

Variation: substitute aspects of other types of work, e.g. coursework, practical work, etc. in which the student may have successful experience and which can be seen as analogous to specific aspects of project work.

---ooOoo---

So: try to change the student's perception of their ability by getting them to reflect on previous successful experience.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

9.4 This is for real

Some students are unmotivated because they regard academic work as artificial, and academic staff as lacking in "real world" skills.

---ooOoo---

This bundle bridges the gap between university and industry/commerce by using "real life" trainers to teach the key skills necessary for good project work.

How it works is you get staff from the training departments of companies with a large IT base to provide learning opportunities in non-technical skills such as presentation skills, report writing, team skills, project management or time management.

Because the sessions are run by experts in the field of training for these key skills, students accept their importance, and their real world relevance, more readily than they would from academic staff. Useful company contacts are also made.

It works better if key skills are assessed as part of the project and if the sessions are scheduled such that the students need to use their newly acquired skills soon afterwards.

It doesn't work if companies misunderstand the purpose of the session, so close liaison is necessary.

---ooOoo---

So: help students see the relevance of the processes and production of University projectwork by involving people from the "real" world in projects.



From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

9.5 Get to know them

Motivation can be associated with feelings of belonging and security but students are often allocated a project supervisor who, being both an authority figure and a complete stranger, is to be distrusted.

---ooOoo---

This bundle recognises that when students, especially adults, sign up for a course they do not do so to fulfil the desires of their teachers; students have their own objectives in taking a course. This is why we must get to know them. This bundle also recognises that student motivation can be altered as much by general encouragement and support as by specific intellectual guidance and academic intervention.

How it works is if you judge that the student's motivation would be helped by a closer relationship, find out about the student before the first meeting. Consult students' academic records and talk to students' tutors or colleagues who have taught them. At the first meeting make a point of introducing yourself?don't stick to professional issues, let them know you're human and have other interests. If it is possible, meet project students socially.

This works better if staff-student relations generally are friendly and if teaching staff take an interest in individual students.

It doesn't work if students perceive insincerity or if they have a chip on their shoulder about academics and the "real world". Students must understand that good relationships are no substitute for quality work.

---ooOoo---

So: help students to feel they belong.

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>

9.6 Here's one I prepared earlier

For some students a lack of motivation can derive from a lack of confidence in their ability to use certain tools or techniques.

---ooOoo---

This bundle gives students the opportunity to boost their confidence by practising the skills required for their project in a way that does not affect their marks.

How it works is you take a previously completed project and ask the student to make a relatively small alteration to some part of it in such a way that they have the opportunity to practice using the relevant tool or technique. This enables the students to become more confident in using the software tools and methods of software development that they will later be required to use in their own work.

This works better if it requires a short amount of time and effort in comparison with the main project work and if it can be done concurrently with other aspects of the main project.

It doesn't work unless the previous project has been carefully designed and executed to match the learning goals. This can be a problem in the first instance, but once developed it can be re-used and/or enhanced in subsequent years.

---ooOoo---

So: build their confidence using practice exercises.



How to write this book

You have good ideas which others may find useful, but how will you make them available?

---ooOoo---

This bundle provides a mechanism for the capture and presentation of good practice in Computing projectwork.

The way it works is having seen the way that pieces of practice are abstracted from their context to make them easy for other practitioners to adopt, use the form of a bundle to identify and reflect on your own good practices. This is in itself useful.

If you think in terms of the phrases which introduce each section of a bundle they should act as levers to help extract the core features of your practice from its context - as well as making them visible and more accessible to others.

If you then write them down, you have a bundle which can be shared.

It works better if the title is relevant and memorable. **It works better if** the problem statement rings bells with your audience and the solution statement really is a solution to that problem. **It also works better if** the body of your bundle gives enough detail to give the reader confidence that they can use your practice, but not so much that it binds the solution to your context only.

It doesn't work if you don't then publish your bundle.

---ooOoo---

So: use this format to capture your practice and put it on the web page: <http://www.cs.kent.ac.uk/national/EPCOS>

From: Sally Fincher, Marian Petre & Martyn Clark *Computer Science Projectwork: Principles and Pragmatics*, Springer-Verlag 2001. This "Field Edition" of Part Two available from: <http://www.cs.kent.ac.uk/national/EPCOS>