Chapter 10

Managing contracts

OBJECTIVES

When you have completed this chapter, you will be able to:

- understand the advantages and disadvantages of using goods and services brought in from outside the organization;
- distinguish among the different types of contract;
- follow the stages needed to negotiate an appropriate contract;
- outline the contents of a contract for goods and services;
- plan the evaluation of a proposal or product;
- administer a contract from its signing until the final acceptance of project completion.

10.1 Introduction

In the Brightmouth College scenario, the management of the college have made a decision to obtain their software from an external supplier. Given the range of payroll software on the market and their own limited capability for developing new and reliable software, this would seem sensible. Meanwhile at IOE, Amanda has available, at least in theory, a team of software developers who are employees of IOE. However, the demand for software design and construction effort will fluctuate, rising when a new project is initiated and trailing off as it is completed. In-house developers could thus have periods of intense pressure when new projects are being developed, interspersed by periods of relative idleness. The IOE management might therefore decide that it would be more cost-effective to get an outside software house to carry out the new development while a reduced group of in-house software development staff remain busy maintaining and giving support to the users of existing systems.

It is not unusual for a major organization to spend 6 to 12 months and 40% of the total acquisition and implementation budget on package evaluation with major customer service and support applications (Demian Martinez, Decision Drivers Inc., *Computing*, 23 July 1998).

The buying in of both goods and services, rather than 'doing it yourself', is attractive when money is available but other, less flexible, types of resource, especially staff time, are in short supply. However, there are hazards for organizations who adopt this policy. Many of these potential dangers arise from the fact that considerable staff time and attention will still be needed to manage a contracted out project successfully. Although the original motivation for contracting out might have been to reduce management effort, it is essential that customer organizations such as Brightmouth College and IOE find time to make clear their exact requirements at the beginning of the planned work, and also to ensure that the goods and services that result are in fact what are actually required.

Also, it needs hardly be said that potential suppliers are more likely to be flexible and accommodating before any contract has been signed than they will be afterwards – especially if the contract is for a fixed price. All this points to the need for as much forethought and planning with an acquisition project as with an internal development project.

In the remainder of this chapter, we will first discuss the different types of contract that can be negotiated. We will then follow through the general steps that ought to be followed when placing a contract. The issues that ought to be considered when drafting a contract are then examined. We conclude by describing some of the things that need to be done while the contract is actually being executed.

It was, for example, reported that two consortia led by Sema and EDS respectively had spent £4 million over two years bidding for a UK government project to renew the IT infrastructure in the prison service – the final job was estimated as being worth £350 million (*Computing*, 13 August, 1998).

Note that the bargaining position of the customer will be much stronger if their business is going to be very valuable. If you are buying a cut-price computer game from a local store, you are unlikely to be able to negotiate variations on the supplier's standard contract of sale! (Indeed, because of the inequality of the parties in such circumstances, such sales are subject to special consumer protection laws). It is reasonable for potential suppliers to weigh up carefully the time and money they are willing to spend responding to a customer's initial request, as there is no guarantee of their obtaining the final contract.

10.2 Types of contract

The external resources required could be in the form of *services*. A simple example of this could be using temporary staff on short term contracts to carry out some project tasks. At Brightmouth College, Brigette could use temporary staff to type into the computer system the personnel details needed to set up the payroll standing data for the new system, while at IOE a decision might be made to carry out the required system building in-house but to augment the permanent staff with contract programmers for the duration of the project. A more far-reaching use of external services would be for the contractor not only to supply the new system but to also operate it on the customer's behalf. For example, it might well be worth Brightmouth College abandoning the idea of buying a package and instead getting a payroll services agency to carry out all the payroll work on their behalf.

On the other hand, the contract could be placed for the supply of a *completed software application*.

This could be:

- a *bespoke* system, that is, a system that is created from scratch specifically for one customer;
- *off-the-shelf*, which you buy 'as is' – this is sometimes referred to as *shrink-wrapped* software;
- *customized off-the-shelf* (COTS) software – this is a basic core system, which is modified to meet the needs of a particular customer.

Where equipment is being supplied then, in English law, this may be regarded as a contract for the supply of *goods*. In the case of the supply of software this may be regarded as supplying a service (to write the software) or the granting of a *licence* (or permission) to use the software, which remains in the ownership of the supplier. These distinctions will have legal implications.

David Bainbridge's *Introduction to Computer Law*, Pitman, 3e, 1996 is highly recommended as a guide to the legal aspects of IT contracts.

Exercise 10.1

Which of the three system options (that is, bespoke, off-the-shelf or COTS) might Amanda consider with regard to the IOE maintenance group accounts system? What factors would she need to take into account?

Another way of classifying contracts is by the way that the payment to suppliers is calculated. We will look at:

- fixed price contracts;
- time and materials contracts;
- fixed price per delivered unit contracts

The section on ways of assessing supplier payments draws heavily on material from Paul Radford and Robyn Lawrie of Charismatek Software Metrics, Melbourne, Australia.

Fixed price contracts

As the name implies, in this situation a price is fixed when the contract is signed. The customer knows that, if there are no changes in the contract terms, this is the price to be paid on the completion of the work. In order for this to be effective, the customer's requirement has to be known and fixed at the outset. In other words, when the contract is to construct a software system, the detailed requirements analysis must already have been carried out. Once the development is under way, the customer will not be able to change their requirements without renegotiating the price of the contract.

The advantages of this method are the following.

- **Known customer expenditure** If there are few subsequent changes to the original requirements, then the customer will have a known outlay.
- **Supplier motivation** The supplier has a motivation to manage the delivery of the system in a cost-effective manner.

The disadvantages include the following.

- **Higher prices to allow for contingency** The supplier absorbs the risk for any errors in the original estimate of product size. To reduce the impact of this risk, the supplier will add a margin when calculating the price to be quoted in a tender.
- **Difficulties in modifying requirements** The need to change the scope of the requirements sometimes becomes apparent as the system is developed – this can cause friction between the supplier and customer.
- **Upward pressure on the cost of changes** When competing against other potential suppliers, the supplier will try and quote as low a price as possible. If, once the contract is signed, further requirements are put forward, the supplier is in a strong position to demand a high price for these changes.
- **Threat to system quality** The need to meet a fixed price can mean that the quality of the software suffers.

Time and materials contracts

With this type of contract, the customer is charged at a fixed rate per unit of effort, for example, per staff-hour. At the start of the project, the supplier normally provides an estimate of the overall cost based on their current understanding of the customer's requirements, but this is not the basis for the final payment.

The advantages of this approach are the following.

- **Ease of changing requirements** Changes to requirements are dealt with easily. Where a project has a research orientation and the direction of the project changes as options are explored, then this can be an appropriate method of calculating payment.
- **Lack of price pressure** The lack of price pressure can allow better quality software to be produced.

The disadvantages of this approach are:

- **Customer liability** The customer absorbs all the risks associated with poorly defined or changing requirements.
- **Lack of incentives for supplier** The supplier has no incentive to work in a cost-effective manner or to control the scope of the system to be delivered.

Because the supplier appears to be given a blank cheque, this approach does not normally find favour with customers. However, the employment of contract development staff, in effect, involves this type of contract.

Fixed price per unit delivered contracts

Function point counting was discussed in Chapter 5.

This is often associated with function point (FP) counting. The size of the system to be delivered is calculated or estimated at the outset of the project. The size of

the system to be delivered might be estimated in lines of code, but FPs can be more easily and reliably derived from requirements documents. A price per unit is also quoted. The final price is then the unit price multiplied by the number of units. Table 10.1 shows a typical schedule of prices.

Table 10.1 *A schedule of charges per function point*

Function point count	*Function design cost per FP*	*Implementation cost per FP*	*Total cost per FP*
Up to 2,000	$242	$725	$967
2,001–2,500	$255	$764	$1,019
2,501–3,000	$265	$793	$1,058
3,001–3,500	$274	$820	$1,094
3,501–4,000	$284	$850	$1,134

This Table comes from D. Garmus and D. Herron, *Measuring The Software Process*, Prentice Hall, 1996.

The company who produced this table, RDI Technologies of the USA, in fact charge a higher fee per FP for larger systems. For example, a system to be implemented contains 2,600 FPs. The overall charge would be 2,000 × $967, plus 500 × $1,1019, plus 100 × $1,058.

One problem that has already been noted is that the scope of the system to be delivered can grow during development. The software supplier might first carry out the system design. From this design, an FP count could be derived. A charge could then be made for design work based on the figures in the 'Function design cost per FP' column. This, if the designed system was counted at 1,000 FPs, would be 1000 × $242 = $242,000. If the design were implemented, and the actual software constructed and delivered, then the additional 1000 × $725 = $725,000 would be charged. If the scope of the system grows because the users find new requirements, these new requirements would be charged at the combined rate for design and implementation. For example, if new requirements amounting to 100 extra FPs were found, then the charge for this extra work would be $967 × 100 = $96,700.

Exercise 10.2

A system to be designed and implemented is counted as comprising 3,200 FPs. What would be the total charge according to the schedule in Table 10.1?

The advantages of this approach are as follows.

- **Customer understanding** The customer can see how the price is calculated and how it will vary with changed requirements.
- **Comparability** Pricing schedules can be compared.
- **Emerging functionality** The supplier does not bear the risk of increasing functionality.

- **Supplier efficiency** The supplier still has an incentive to deliver the required functionality in a cost-effective manner (unlike with time and materials contracts).
- **Life cycle range** The requirements do not have to be definitively specified at the outset. Thus the development contract can cover both the analysis and design stages of the project.

The disadvantages of this approach are as follows.

The impact of late changes will be further discussed in Chapter 12 on Software Quality.

- **Difficulties with software size measurement** Lines of code can easily be inflated by adopting a verbose coding style. With FPs, there can be disagreements about what the FP count should really be: in some cases, FP counting rules might be seen as unfairly favouring either the supplier or customer. Users, in particular, will almost certainly not be familiar with the concept of FPs and special training might be needed for them. The solution to these problems might be to employ an independent FP counter.
- **Changing requirements** Some requested changes might affect existing transactions drastically but not add to the overall FP count. A decision has to be taken about how to deal with these changes. A change made late in the development cycle will almost certainly require more effort to implement than one made earlier.

To reduce the last difficulty, one suggestion from Australia has been to vary the charge depending on the point at which they have been requested – see Table 10.2.

The table comes from the draft Acquisition of customized software policy document, published by the Department of State Development, Victoria, 1996.

Table 10.2 *Examples of additional charges for changed functionality*

	Pre-acceptance testing handover	*Post-acceptance testing handover*
Additional FPs	100%	100%
Changed FPs	130%	150%
Deleted FPs	25%	50%

Exercise 10.3

A contract stipulates that a computer application is to be designed, constructed and delivered at a cost of $600 per FP. After acceptance testing, the customer asks for changes to some of the functions in the system amounting to 500 FPs and some new functions which amount to 200 additional FPs. Using Table 10.2, calculate the additional charge.

In addition to the three payment methods above, there are other options and permutations of options. For instance, the implementation of an agreed specification could be at a fixed price, with provision for any additions or changes to the requirements to be charged for on a per FP basis. Another example could be where the contractor has to buy in large amounts of equipment, the price of which

might fluctuate through no fault of the contractor. In this case it is possible to negotiate a contract where the final price contains a fixed portion for labour plus an amount that depends on the actual cost of a particular component used.

Exercise 10.4

It is easy to see why passing on fluctuations in equipment costs can be advantageous to the contractor. However, is there any advantage to the customer in such an arrangement?

An underlying problem with software can once again be seen when the questions of contractual obligations and payment are considered – namely its relative invisibility and its flexibility. These mean that system size and consequently development effort tends to be very difficult to judge. If contractors are realistically to quote firm prices for work, then the tasks they are asked to undertake must be carefully constrained. For example, it would be unrealistic for a contractor to be asked to quote a single price for all the stages of a development project: how can they estimate the construction effort needed when the requirements are not yet established? For this reason it will often be necessary to negotiate a series of contracts, each covering a different part of the system development life cycle.

The concept of 'IS adaptations' in Euromethod (see Appendix C) supports the idea of a series of separate contracts to implement a single system.

Another way of categorizing contracts, at least initially, is according to the approach that is used in contractor selection:

- open;
- restricted;
- negotiated.

This categorization is based on European Union regulations.

Open tendering process

In this case, any supplier can bid to supply the goods and services. Furthermore all bids that are compliant with the original conditions laid down in the *invitation to tender* must be considered and evaluated in the same way as all the others. With a major project where there are lots of bids and the evaluation process is time-consuming, this can be an expensive way of doing things.

In recent years, there has been a global movement towards removing artificial barriers that hamper businesses in one country supplying goods and services in another. Examples of this are the drives by bodies such GATT and the European Union to ensure that national governments and public bodies do not limit the granting of contracts to their fellow nationals without good reason. One element of this is the laying down of rules about how tendering processes should be carried out. In certain circumstances, these demand that an open tendering process be adopted.

GATT stands for 'General Agreement on Trade and Tariffs'. The specific part of GATT that is relevant here is the Agreement on Government Procurement. GATT covers the EU and also several other countries including the US and Japan.

Restricted tendering process

In this case, there are bids only from suppliers who have been invited by the customer. Unlike the open tendering process, the customer may at any point reduce the number of potential suppliers being considered. This is usually the best approach to be adopted. However, it is not without risk: where the resulting contract is at a fixed price, the customer assumes responsibility for the correctness and completeness of the requirements specified to the prospective suppliers. Defects in this requirements documentation sometimes allow a successful bidder subsequently to claim for additional payments.

Negotiated procedure

There are often, however, some good reasons why the restricted tendering process might not be the most suitable in some particular sets of circumstances. Say, for example, that there is a fire that destroys part of an office, including IT equipment. The key concern here is to get replacement equipment up and running as quickly as possible and there is no time to embark on a lengthy tendering process. Another situation might be where a new software application had been successfully built and implemented by an outsider, but the customer decides to have some extensions to the system. As the original supplier has staff who have complete familiarity with the existing system, it might once again be inconvenient to approach other potential suppliers via a full tendering process.

In these cases, an approach to a single supplier might be justified. However, it takes little imagination to realise that approaching a single supplier can open the customer up to charges of favouritism and should be done only where there is a clear justification.

10.3 Stages in contract placement

We are now going to discuss the typical stages in awarding a contract.

Requirements analysis

This discussion assumes that a feasibility study has already provisionally identified the need for the intended software.

Before potential supplier can be approached, you need to have a clear set of requirements. Two points need to be emphasized here. The first is that it is easy for this step to be skimped where the user has many day-to-day pressures and not much time to think about future developments. In this situation, it can be useful to bring in an external consultant to draw up a requirements document. Even here, users and their managers need to look carefully at the resulting requirements document to ensure that it accurately reflect their needs. As David Bainbridge has pointed out: '*the lack of, or defects in, the specification are probably the heart of most disputes resulting from the acquisition of computer equipment and software*'

David Bainbridge *ibid* page 135.

The requirements document might typically have sections with the headings shown in Table 10.3. This requirements document is sometimes called an operational requirement or OR.

Table 10.3 *Main sections in a requirements document*

	Section name
1	Introduction
2	A description of any existing systems and the current environment
3	The customer's future strategy or plans
4	System requirements • mandatory • desirable
5	Deadlines
6	Additional information required from potential suppliers

The requirements define carefully the *functions* that need to be carried out by the new application and all the necessary *inputs* and *outputs* for these functions. The requirements should also state any *standards* with which there should be compliance, and the existing systems with which the new system needs to be compatible. As well as these functional requirements, there will also need to be operational and quality requirements concerning such matters as the required response times, reliability, usability and maintainability of the new system.

Chapter 12 on Software Quality discusses how aspects of quality can be measured.

In general, the requirements document should state *needs* as accurately as possible and should avoid technical specifications of possible solutions. The onus should be placed on the potential suppliers to identify the technical solutions that they believe will meet the customer's stated needs. After all, they are the technical experts who should have access to the most up-to-date information about current technology.

Each requirement needs to be identified as being either *mandatory* or *desirable*.

- **Mandatory** If a proposal does not meet this requirement then the proposal is to be immediately rejected. No further evaluation would be required.
- **Desirable** A proposal might be deficient in this respect, but other features of the proposal could compensate for it.

For example, in the case of the Brightmouth College payroll acquisition project, Brigette might identify as a mandatory requirement that any new system should be able to carry out all the processes previously carried out by the old system. However a desirable feature might be that the new payroll software should be able to produce accounting details of staff costs in an electronic format that can be read directly by the college's accounting computer system.

One suggestion is that the weighting between product criteria and supplier criteria when selecting software ought to be 50:50 (Demian Martinez, Decision Drivers Inc., *Computing* 23 July 1998).

Among the other details that should be included in the requirements document to be issued to potential suppliers would be requests for any information needed to help us judge the standing of organization itself. This could include financial reports, references from past customers and the CVs of key development staff.

Evaluation plan

Having drawn up a list of requirements, we now need to draw up a plan of how the proposals that are submitted are to be evaluated. The situation will be slightly different if the contract is for a system that is to be specially written as opposed to an off-the-shelf application. In the latter case, it is the system itself that is being evaluated while in the former situation it is a proposal for a system.

First, a means of checking that all the mandatory requirements have been met needs to be identified. The next consideration is of how the desirable requirements can be evaluated. The problem here is weighing the value of one quality against another. The ISO 9126 standard, which is discussed in the chapter on software quality, can be used to decide that one system has more 'quality' than another, but if there is a difference in price between the two, we need to be able to estimate if the increase in quality is worth the additional price. Hence 'value for money' is often the key criterion. For example, we mentioned above an instance where the existence of an accounting link file was identified as a desirable requirement in the case of the Brightmouth College payroll acquisition project. Could a financial value be placed on this? If we were to cost clerical effort at £20 an hour and we knew that four hours of clerical effort a month went into entering staffing costs into the accounting computer system, then we could conclude that over a four year period (£20 an hour × 4 hours a month × 48 months), or £3840, would be saved. If system A has this feature and costs £1000 more than system B, which does not, then this would seem to give system A an advantage. If, however, system A cost £5000 more than B then the picture would be different.

Some of these issues were touched on in Chapter 3 on project evaluation.

It needs to be stressed that the costs to be taken into account are those for the whole of the lifetime of the proposed system, not just the costs of acquiring the system. Also, where the relationship with the supplier is likely to be ongoing, the supplier organization needs to be assessed as well as its products.

Exercise 10.5

One desirable feature sought in the Brightmouth College payroll is the ability to raise staff to the next point in their salary scale automatically at the beginning of each payroll year. At present, the new scale points have to be input clerically and then be checked carefully. This takes about 20 hours of staff effort each year, which can be costed at £20 an hour. System X has this feature, but system Y does not. System X also has a feature which can automatically produce bar-charts showing payroll expenditure per department. Such a report currently has to be produced twice a year by hand and on each occasion takes about 12 hours effort to complete. With System Y, changes to department names can be carried out without any coding effort, whereas in the case of System X, the supplier would charge a minimum of £300 to do this. The college authorities estimate that there is 50% chance that this could occur during the expected four year lifetime of the system. System X costs £500 more than System Y. On the basis of this information which system appears to give better value for money?

Invitation to tender

Having produced the requirements and the evaluation plan, it is now possible to issue the invitation to tender to prospective suppliers. Essentially, this will be the requirement document with a supporting letter, which may have additional information about how responses to the invitation are to be lodged. A deadline will be specified and it is hoped that by then a number of proposals with price quotations will have been received.

In English law, for a contract to exist there must be an offer on one side that must be accepted by the other side. The invitation to tender is not an offer itself but an invitation for prospective suppliers to make an offer.

Certain new problems now emerge. The requirements that have been laid down might be capable of being satisfied in a number of different ways. The customer needs to know not only a potential supplier's price but also the way in which they intend to satisfy the requirements – this will be particularly important where the contract is to build a new system from scratch.

In some relatively straight-forward cases, it would be enough to have some post-tender clarification and negotiation to resolve issues in the supplier's proposal. With more complex projects a more elaborate approach may be needed. One way of getting the detail of the suppliers' proposals elaborated is to have a two stage tendering process.

This approach is recommended by the CCTA in the United Kingdom.

In the first stage, technical proposals are requested from potential suppliers who do not necessarily quote any prices at this stage. Some of these proposals can be dismissed out of hand as not being able to meet the mandatory requirements. With the remaining ones, discussions may be held with representatives of the suppliers in order to clarify and validate the technical proposals. The suppliers may be asked to demonstrate certain aspects of their proposals. Where short-comings in the proposal are detected, the supplier may be given the opportunity to remedy these.

The result of these discussions could be a *Memorandum of Agreement* (MoA) with each prospective supplier. This is an acceptance by the customer that the proposed solution (which might have been modified during discussions) offered by the supplier satisfactorily meets the customer's requirement.

In the second stage, tenders are invited from all the suppliers who have signed individual Memoranda of Agreement. The tender would incorporate the MoA and would be concerned with the financial terms of a potential contract.

If a design has to be produced as part of the proposal made by a supplier in response to an invitation to tender, then a difficulty would be that the supplier would have to do a considerable amount of detailed design work with only a limited prospect of being paid for it. One way of reducing this burden is for the customer to choose a small number of likely candidates who will be paid a fee to produce design proposals. These can then be compared and the final contract for construction awarded to the most attractive proposal.

The ISO 12207 has a rather different approach. Once a contract for software construction is signed, the supplier develops a design, which then has to be agreed by the customer.

Evaluation of proposals

This needs to be done in a methodical and planned manner. We have already mentioned the need to produce an evaluation plan, which will describe how each proposal will be checked to see if it meets each requirement. This reduces risks of requirements being missed and ensures that all proposals are treated consistently. Otherwise, there is a risk that a proposal might be unfairly favoured because of the presence of a feature that was not requested in the original requirement.

It will be recalled that an application could be either bespoke, off-the-shelf, or customized. In the case of off-the-shelf packages, it would be the software itself that would be evaluated and it might be possible to combine some of the evaluation with acceptance testing. With bespoke development it would be a proposal that is evaluated, while COTS may involve elements of both. Thus different planned approaches would be needed in each case.

The process of evaluation may include:

- scrutiny of the proposal documents;
- interviewing suppliers' representatives;
- demonstrations;
- site visits;
- practical tests.

The proposal documents provided by the suppliers can be scrutinized to see if they contain features satisfying all the original requirements. Clarification might need to be sought over certain points. Any factual statements made by a supplier imply a legal commitment on their part if it influences the customer to offer the contract to that supplier. It is therefore important to get a written, agreed, record of these clarifications. The customer may take the initiative here by keeping notes of meetings and then writing afterwards to the suppliers to get them to confirm the accuracy of the notes. A supplier might, in the final contract document, attempt to exclude any commitment to any representations that have been made in pre-contract negotiations – the terms of the contract need to be scrutinized in this respect.

Where the delivered product is to be based on an existing product it might be possible to see a demonstration. A danger with demonstrations is that they can be controlled by the supplier and as a passive observer it is often difficult to maintain full attention for more than, say, half-an-hour. Because of this, the customer should produce a schedule of what needs to be demonstrated, ensuring that all the important features are seen in operation.

With off-the-shelf software, it should be possible to have actual access to the application. For example, a demonstration version, which closes itself down after 30 days, might be available. Once again a test plan is needed to ensure that all the important features are evaluated in a complete and consistent manner. Once a particular package emerges as the most likely candidate, it needs to be carefully

investigated to see if there are any previously unforeseen factors that might invalidate this choice.

A frequent problem is that while an existing application works well on one platform with a certain level of transactions, it does not work satisfactorily on the platform that the customer has or at the level of throughput that it would be subjected to in the customer's work environment. Demonstrations will not generally reveal this problem. Visits to operational sites already using the system will be more informative in this respect. In the last resort a special volume test could be conducted.

Exercise 10.6

How would you evaluate the following aspects of a proposal?

i. The usability of an existing software application.

ii. The usability of a software application that is yet to be designed and constructed.

iii. The maintenance costs of hardware to be supplied.

iv. The time taken to respond to requests for software support.

v. Training.

Where substantial sums of money are involved, legal advice on the terms of the contract is essential.

Eventually a decision will be made to award the contract to one of the suppliers. One of the central reasons for using a structured and, as far as possible, objective approach to evaluation is to be able to demonstrate that the decision has been made impartially and on merit. In most large organizations, placing a contract involves the participation of a second party within the organization, such as a contracts department, who can check that the correct procedures have been carried out. Also the final legal format of a contract will almost certainly require some legal expertise.

In any case, not only should the successful candidate be notified but the unsuccessful candidates should also be told of the decision. This is not simply a matter of courtesy: under GATT or EU rules, there is a legal requirement to do this in certain circumstances. It makes dealing with unsuccessful bidders easier if they can be given clear and objective reasons why their proposals did not find favour.

10.4 Typical terms of a contract

In a textbook such as this, it is not possible to describe all the necessary content of contracts for IT goods or services. It is possible, however to outline some of the major areas of concern.

Definitions

The terminology used in the contract document may need to be defined, for example, who is meant by the words 'client' and 'supplier'.

Form of agreement

For example, is it a contact of sale, a lease, or a licence? Also, can the subject of the contract, such as a licence to use a software application, be transferred to another party?

Goods and services to be supplied

Equipment and software to be supplied This includes an actual list of the individual pieces of equipment to be delivered, complete with the specific model numbers.

Services to be provided This covers such things as:

- training;
- documentation;
- installation;
- conversion of existing files;
- maintenance agreements;
- transitional insurance arrangements.

Ownership of the software

Who has ownership of the software? There are two key issues here: firstly, whether the customer can sell the software to others and, secondly, whether the supplier can sell the software to others. Where off-the-shelf software is concerned, the supplier often simply grants a license for you to use the software. Where the software is being written specially for a customer, then that customer will normally wish to ensure exclusive use of the software – they may object to software which they hoped would give them a competitive edge being sold to their rivals. They could do this by acquiring the copyright to the software outright or by specifying that they should have *exclusive use* of the software. This would need to be written into the contract. Where a core system has been customized by a supplier, then there is less scope for the customer to insist on exclusive use.

Where software is written by an employee as part of a contract of employment, it is assumed that the copyright belongs to the employer. Where the customer organization has contracted an external supplier to write software, the contract needs to make clear who is going to retain the copyright – it cannot, in this case, be automatically assumed it is the customer. The customer might have decided to take over responsibility for maintenance and further development once the software is delivered and in this case will need access to the source code. In other

cases, where the customer does not have an adequate in-house maintenance function, the supplier can retain the source code, and the customer will have to approach the supplier for any further changes. There are many potential dangers with this, not the least being that the supplier could go out of business. An escrow agreement can be included in the contract so that up-to-date copies of the source code are deposited with a third party. In the United Kingdom, the National Computing Centre provide an escrow service.

Environment

Where physical equipment is to be installed, the demarcation line between the supplier's and customer's responsibilities with regard to such matters as accommodation and electrical supply needs to be specified. Where software is being supplied, the compatibility of the software with the existing hardware and operating system platforms would need to be confirmed.

Customer commitments

Even when work is carried out by external contractors, a development project still needs the participation of the customer. The customer will have to provide accommodation for the suppliers and perhaps other facilities such as telephone lines.

Acceptance procedures

Good practice would be to accept a delivered system only after it has undergone user acceptance tests. This part of the contract would specify such details as the time that the customer will have to conduct the tests, deliverables upon which the acceptance tests depend and the procedure for signing off the testing as completed.

Some customers find that specially written or modified software is not thoroughly tested by the supplier before delivery. Some suppliers seem to think that it is cheaper to get the customer to do the testing for them!

Standards

This covers the standards with which the goods and services should comply. For example, a customer can require the supplier to conform to the ISO 12207 standard relating to the software life cycle and its documentation (or, more likely, a customized sub-set of the standard). Within the European Union, government customers with contracts for projects above a certain threshold value must, by law, ensure that the work conforms to certain standards.

Project and quality management

The arrangements for the management of the project must be agreed. Among these would be frequency and nature of progress meetings and the progress information to be supplied to the customer. The contract could require that appropriate ISO 9000-series standards be followed. The ISO 12207 standard provides for the customer to have access to quality documentation generated internally by the supplier, so that the customer can ensure that there is adherence to standards.

Timetable

This provides a schedule of when the key parts of the project should be completed. This timetable will commit both the supplier and the customer. For example, the supplier might be able to install the software on the agreed date only if the customer makes the hardware platform available at that point.

Price and payment method

Obviously the price is very important! What also needs to be agreed is when the payments are to be made. The supplier's desire to be able to meet costs as they are incurred needs to be balanced by the customer's requirement to ensure that goods and services are satisfactory before parting with their money.

Miscellaneous legal requirements

This is the legal small print. Contracts often have clauses that deal with such matters the legal jurisdiction that will apply to the contract, what conditions would apply to the sub-contracting of the work, liability for damage to third parties, and liquidated damages. *Liquidated damages* are estimates of the financial losses that the customer would suffer if the supplier were to fall short of their obligations. It is worth noting that under English law, the penalties laid down in penalty clauses must reflect the actual losses the customer would suffer and cannot be unrealistic and merely punitive. Even this limitation will not be enough in some cases as far as the supplier is concerned. As computer systems assume increasingly critical roles in many organizations and in safety-critical systems can even be life-threatening in the case of malfunction, the possible consequential damage could be very great. Suppliers will not unnaturally try to limit this kind of liability. The courts (in England and Wales) have tended to look critically at such attempts at limiting liability, so that suppliers will, in the case of major contracts, take out insurance to cover such liabilities.

If there is a dispute, resorting to litigation, while being lucrative to the lawyers involved, is both time-consuming and expensive. An alternative is to agree that disputes be settled by *arbitration*. This requires that any dispute be referred to an expert third party whose decision as to the facts of the case is binding. Even this procedure is seldom quick and inexpensive and another option is *alternative dispute resolution* where a third party acts as a mediator who has only an advisory capacity and attempts to broker an agreement between the two sides.

10.5 Contract management

Euromethod offers guidance about how decision points can be planned.

We now need to consider the communications between the supplier and the customer while the work contracted for is being carried out. It would probably suit all concerned if the contractor could be left to get on with the work undisturbed. However, at certain *decision points*, the customer needs to examine work already done and make decisions about the future direction of the project. The project will

require representatives of the supplier and customer to interact at many points in the development cycle – for example, users need to be available to provide information needed to carry out effective detailed interface design.

This interaction, or other external factors, often leads to changes being needed, which effectively vary the terms of the contract and so a careful change control procedure is needed. Each of these topics will now be tackled in a little more detail.

When a the contract is being negotiated, certain key points in the project can be identified where customer approval is needed before the project can proceed. For example, a project to develop a large system can be divided into increments. For each increment there could be an interface design phase, and the customer needs to approve the designs before the increment is built. There could also be a decision point between increments.

Chapter 4 discusses incremental delivery.

For each decision point, the deliverables to be presented by the suppliers, the decisions to be made by the customer and the outputs from the decision point all need to be defined. These decision points have added significance if payments to the supplier are based on them. Not only the supplier but also the customer has responsibilities with respect to these decision points – for example, the supplier should not be unnecessarily delayed while awaiting customer approval of some interim deliverable.

Where work is contracted out there will be a general concern about the quality of that work. The ISO 12207 standard envisages the possibility of there being agents, employed independently of the supplier or customer, who will carry out verification, validation and quality assurance. It also allows for joint reviews of project processes and products, the nature of which needs to be clearly agreed when the contract is negotiated, otherwise the supplier might claim unwarranted interference in their work.

As the system is developed a need to change certain of the requirements often emerges. As noted earlier, essentially, this is varying the terms of the contract. Oral evidence is not normally admissible to contradict, add to, or vary the terms of a written contract, so that agreed changes need to be documented properly. An effective change control procedure is therefore needed to record requests for changes, along with the supplier's agreement to them and any fees for the additional work.

It could happen that the supplier does not meet one or more of their legal obligations. This might be through no fault of theirs, if, for example, the customer has caused the delay by being tardy in giving the necessary approvals for intermediate products. If no action is taken when the default occurs, this can be taken to imply that the customer in fact condones the failure and this can lead to the loss of a right to legal redress. The customer should therefore protect their legal rights by officially notifying the supplier as soon as possible that the failure has been recognized. It will be recalled that under English law any claim for liquidated damages should be based on actual losses. From the point where the default occurs, the customer needs to keep an accurate record of the actual losses incurred as a result of the default including any consequential losses.

10.6 Acceptance

When the work has been completed, the customer needs to take action to carry out acceptance testing. The contract might put a time limit on how long acceptance testing can take, so the customer must be organized to carry out this testing before the time limit for requesting corrections expires.

We have already noted that some software houses are rather cursory with their pre-acceptance testing: the implication seeming to be that they would rather the users spent their time on testing than they themselves. This imposition can be reduced by asking to approve the supplier's internal test plans. An associated pitfall is that once the main development work is completed, the supplier not unnaturally wants to reallocate the most productive staff to other projects. The customer can find that all their problem reports are being dealt with by relative junior members of the supplier's staff, who might not be familiar with all aspects of the delivered system.

Part or all of the payment to the supplier will depend on this acceptance testing. Sometimes part of the final payment will be retained for a period of operational running and is eventually paid over if the levels of reliability are as contracted for. There is usually a period of warranty during which the supplier should fix any errors found for no charge. The supplier might suggest a very short warranty period of say 30 days. It is in the customer's interests to negotiate a more realistic period of say at least 120 days.

10.7 Summary

Some of the key points in this chapter have been:

- the successful contracting out of work requires considerable amounts of management time;
- it is easier to gain concessions from a supplier before a contract is signed than afterwards;
- alternative proposals need to be evaluated as far as possible by comparing costs over the whole lifetime of the system rather than just the acquisition costs;
- a contract will place obligations on the customer as well as the supplier;
- contract negotiation should include reaching agreement on the management of the supplier-customer relationship during the execution of the project.

10.8 Further exercises

1. At IOE, the management are considering 'out-sourcing' the maintenance accounting system, that is, getting an outside specialist organization to take over the operation, maintenance, and support activities associated with the

system. Write a short memorandum to management outlining the advantages and disadvantages of such a re-organization.

2. In each of the following cases discuss whether the type of application software to be adopted would be most likely to be bespoke, off-the-shelf or COTS.

   (a) A college requires a student fees application. It is suggested that the processes required in the application are similar to those of any billing system with some requirements that are peculiar to the administration of higher education.
   (b) A computer-based application is needed at IOE to hold personnel details of staff employed.
   (c) A national government requires a system that calculates, records and notifies individual tax-payers about income tax charges.
   (d) A hospital needs a knowledge-based system to diagnose the causes of eye complaints.

3. The schedule of charges per function point shown in Table 10.1 has higher rates for larger systems. Give arguments explaining why this might be justified and also arguments against.

4. Table 10.2 has a charge of 25% and 50% of the normal rate for deleting transactions from an application. This seems to be rather high for simply removing code. What work would be involved in deleting functionality that could justify this cost?

5. Assume that IOE has decided on a COTS solution that will replace the whole of the existing maintenance accounting system rather than simply plugging in additional modules to deal with groups accounts. Write a memorandum that Amanda could send to IOE's legal department outlining the important provisions that a contract to supply this system should have.

Chapter 11

Managing people and organizing teams

OBJECTIVES

When you have completed this chapter, you will be able to:

- identify some of the factors that influence people's behaviour in a project environment;
- select the most appropriate people for a project;
- understand the role of continuing training and learning;
- increase staff motivation;
- improve group working;
- use the most appropriate leadership styles;
- understand the characteristics of the various team structures that can be employed.

11.1 Introduction

We are going to examine some of the problems that Amanda and Brigette could meet when dealing with the staff who will be working for them. Where possible we will see if the findings of researchers can provide any ideas about what to do.

First, we will look at some aspects of organizational behaviour (OB) research. There will be three concerns: staff selection, staff development and, which will be dealt with in more detail, staff motivation.

We will look at how the project leader can encourage effective group working and decision making while balancing this, where needed, by purposeful leadership. The final part of the chapter looks at some of the more formal aspects of organizational structures.

The issues raised in this chapter have impacts at all stages of project planning and execution but in particular at the following points (see also Figure 11.1):

- although perhaps having little control over organizational structure, the project leader needs to be aware of its implications (Step 2);
- the scope and nature of activities can be set in a way that will enhance staff motivation (Step 4);
- many risks to project success relate to staffing (Step 6);
- the qualities of individual members of staff should be taken into account when allocating staff to activities (Step 7).

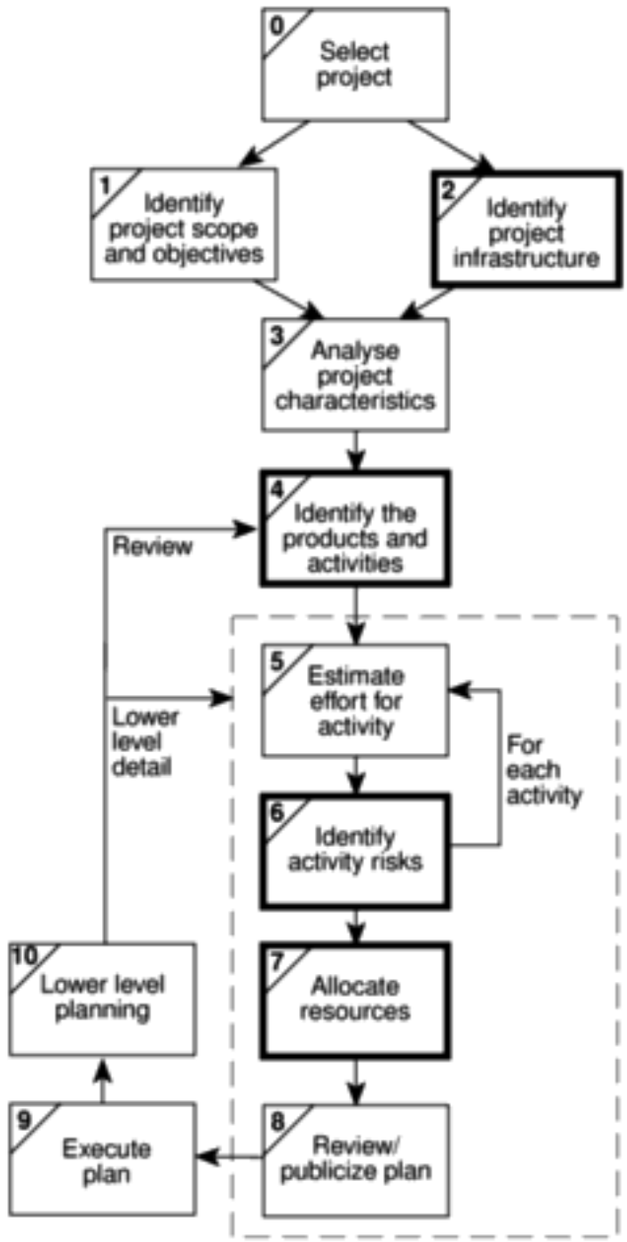

Figure 11.1 *Some places in the Step Wise framework where staffing concerns are important.*

11.2 Understanding behaviour

People with practical experience of working on projects invariably identify the handling of people as one of the most important aspects of project management. What people like Amanda and Brigette will want to know is whether the effective and sensitive management of staff comes only from experience or whether guidance can be usefully sought from writers on the topic.

The field of social science known as organizational behaviour (OB) helps. This has evolved theories that try to explain people's behaviour and that tend to be structured 'If A is the situation then B is likely to result'. Attempts are then made to observe behaviour or to conduct experiments where variables for A and B are measured and a statistical relationship between the two variables is sought. Unlike physical science, it is rarely, if ever, the case that it can be said that B must always follow A.

A major problem is that in the real world there is bound to be a very wide range of influences on a situation, many of which will not be apparent to the observer. It is therefore difficult to decide which set of research findings is relevant. A danger is that we end up with a set of maxims that are little better than superstitions. However, it is to hoped that by examining these questions people can become more sensitive and thoughtful about the problems involved.

In what follows, we will be making references to workers in the OB field such as Taylor, Mayo and McGregor. Rather than overwhelming the reader with references, we recommend the reader who is interested in exploring this topic further to look at Charles Handy's book. Where we have given references these tend to be for works related specifically to an IT environment.

Charles Handy, *Understanding organizations*, 4th edition, Penguin, 1993.

11.3 Organizational behaviour: a background

The roots of studies in OB can be traced back to work done in the late 19th and early 20th centuries by Frederick Taylor. By studying the way that manual workers did tasks, he attempted to work out the most productive way of doing these tasks. The workers were then trained to do the work in this way.

Taylor had three basic objectives:

- to select the best person for the job;
- to instruct such people in the best methods;
- to give incentives in the form of higher wages to the best workers.

Frederick Winslow Taylor, 1856–1915, is regarded as the father of 'scientific management' of which OB is a part.

'Taylorism' is often represented as crude and mechanistic these days. Interestingly though, the Taylorist approach is one that is adopted, in part, in modern sports coaching. A coach will attempt to get a javelin thrower, for example, to throw a javelin in a very exact manner in order to get the maximum effect. In the more mundane world of software development, the growth of

structured methods is an example of this emphasis on best practice. Both Amanda and Brigette will be concerned that tasks are carried out in the proper way. As we will see, more contentious is Taylor's emphasis on the exclusively financial basis of staff motivation, although Amanda and Brigette will be sure to find many colleagues who hold Taylor's view on the importance of 'performance-related pay'. Unfortunately, Amanda and Brigette are likely to have very little control over the financial rewards of their staff. However, they should be encouraged by findings that motivation rests not just on such rewards.

Elton Mayo and his colleagues did this research at the Hawthorne Works of Western Electric in Chicago, hence the 'Hawthorne Effect'.

During the 1920s, OB researchers discovered, while carrying out a now famous set of tests on the conditions under which staff worked best, that not only did a group of workers for whom conditions were improved increase their work-rates, but a control group, for whom conditions were unchanged, also increased their work-rates. Simply showing a concern for what workers did increased productivity. This illustrated how the state of mind of workers influenced their productivity.

The cash-oriented view of work of some managers can thus be contrasted with a more rounded vision of people in their place of work. The two attitudes were labelled *Theory X* and *Theory Y* by Donald McGregor.

Theory X holds that:

- the average human has an innate dislike of work;
- there is a need therefore for coercion, direction and control;
- people tend to avoid responsibility.

Theory Y, on the other hand, holds that:

A 'reward' does not have to be a financial reward – it could be something like a sense of achievement.

- work is as natural as rest or play;
- external control and coercion are not the only ways of bringing about effort directed towards the company's ends;
- commitment to objectives is a function of the rewards associated with their achievement;
- the average human can learn to accept and further seek responsibility;
- the capacity to exercise imagination and other creative qualities is widely distributed.

One way of judging whether a manager espouses Theory X or Theory Y is to observe how the manager's staff react when the boss is absent: if there is no discernible change, then this is a Theory Y environment; if everyone visibly relaxes, it is a Theory X environment. McGregor's distinction between the two theories also draws attention to the way that expectations influence behaviour. If a manager (or teacher) assumes that you are going to work diligently and do well, then you are likely to try and meet these expectations.

11.4 Selecting the right person for the job

Taylor stressed the need for the right person for the job. Many factors, such as the use of software tools and methodologies, affect programming productivity. However, one of the biggest differences in software development performance is among individuals. As early as 1968, a comparison of experienced professional programmers working on the same programming task found a ratio, in one case, of 1:25 between the shortest and longest time to code the program and, more significantly perhaps, of 1:28 for the time taken to debug it. Amanda and Brigette should therefore be rightly concerned to get the best possible people working for them.

B. W. Boehm considered the quality of staff the most important influence on productivity when constructing the COCOMO software cost models (Chapter 5).

What sort of characteristics should they be looking for? Should they go, for example, for the experienced programmer or the new graduate with the first class mathematics degree? It is extremely dangerous to generalize but looking specifically at behavioural characteristics, the American researcher Cheney found that the most important influence on programmer productivity seemed to be experience. Mathematical aptitude had quite a weak influence in comparison.

P. M. Cheney 'Effects of Individual Characteristics, Organizational Factors and Task Characteristics on Computer Programmer Productivity and Job Satisfaction' in *Information and Management*, 7 (1984).

Amanda and Brigette will want staff who can communicate well with each other and, more importantly, with users. They well have some difficulties here. The American researchers Couger and Zawacki found that computing people would appear to have much weaker 'social needs' than people in other professions. They quote Gerald Weinberg: '*If asked, most programmers probably say they prefer to work alone where they wouldn't be disturbed by other people.*' This is reflected in the problem that people attracted to writing software, and are good at it, will not make good managers later in their careers.

J. D. Couger and R. A. Zawacki 'What motivates DP Professionals?' in *Datamation*, 24 (1978).

The recruitment process

Although this is an important matter, it has to be stressed that often project leaders have little choice about the people who will make up the teams – they have to make do with the 'materials that are to hand'. Recruitment might very well be regarded as an organizational responsibility: you might be recruiting someone who will, over a period of time, work in many different parts of the same organization.

Meredith Belbin usefully distinguishes between *eligible* and *suitable* candidates. An eligible candidate is one whose CV (curriculum vitae or résumé) shows, for example, the 'right' number of years in some previous post and the 'right' paper qualifications. Suitable candidates are those who can actually do the job well. An easy mistake is to select an eligible candidate who is not in fact suitable. Suitable candidates who are not technically eligible can, on the other hand, be ideal candidates because, once in post, they are more likely to remain loyal to the organization. Belbin suggests that selection methods that centre on the assessment of actual skills rather than past experience and also a willingness to provide training to make good minor gaps in expertise can be a more effective way of placing suitable staff. It also seems to us to show that policies that avoid discrimination on the grounds of race, gender, age or irrelevant disabilities can be not just socially responsible but also a shrewd recruitment policy.

R. Meredith Belbin, *Team Roles At Work*, Butterworth-Heinemann, 1993

A general approach might be the following.

- **Create a job specification** Advice is needed, as there will be legal implications in an official document. However, formally or informally, the requirements of the job should be documented and agreed.
- **Create a job holder profile** Using the job specification, a profile of the person needed to carry out the job is constructed. The qualities, qualifications, education and experience required would be listed.
- **Obtain applicants** Typically, an advertisement would be placed, either within the organization or outside in the trade or local press. The job holder profile would be examined carefully to identify the medium most likely to reach the largest number of potential applicants at least cost. For example, if a specialist is needed it would make sense to advertise in the relevant specialist journal. The other principle is to give enough information in the advertisement to allow an element of self-elimination. By giving the salary, location, job scope and any essential qualifications, the applicants will be limited to the more realistic candidates.
- **Examine CVs** These should be read carefully and compared to the job holder profile – nothing is more annoying for all concerned than when people have CVs which clearly indicate that they are not eligible for the job and yet they are called for interview.
- **Interviews etc.** A number of different selection techniques can be tried, including aptitude tests, personality tests, and the examination of samples of previous work. All these methods must be related to specific qualities detailed in the job holder profile. Interviews are the most commonly used method. It is better if there is more than one interview session with an applicant and with each session there should not be more than two interviewers because a greater number reduces the possibility of follow-up questions and discussion. Some formal scoring system for the qualities being judged should be devised and interviewers should then decide scores individually which are then compared. An interview should be of a technical nature where the practical expertise of the candidate is assessed, or of a more general nature if not. In the latter case, a major part of the interview will in fact be evaluating and confirming what was stated in the CV – for example any time gaps in the education and employment history would be investigated, and the precise nature of jobs previously done would need to be explored.
- **Other procedures** References will need to be taken up where necessary, and a medical examination might be needed.

Exercise 11.1

A new analyst/programmer is to be recruited to work in Amanda's team at IOE. The intention is to recruit someone who already has some experience. Make a list

of the types of activities that the analyst/programmer should be capable of carrying out that can be used as the basis for a job specification.

11.5 Instruction in the best methods

This is the second concern that we have taken from Taylor. Obviously, there is a difference between loading pig iron (one of Taylor's studies) and writing C programs, but the principle of having established methods and procedures is, we hope, as well understood in software development as in steel-making.

When a new member of the team is recruited, the team leader will need to plan that person's induction into the team very carefully. Where a project is already well under way, this might not be easy. However, the effort should be made – it should pay off eventually as the new recruit will become a fully effective member of the team more quickly.

The need to take into account the time needed to acclimatize new staff was stressed in Chapter 8 on resource allocation.

The team leader should also be aware of the need to assess continually the training needs of their team members. Just as you formulate a user requirement before considering a new system, and you construct a job holder profile before recruiting a member of staff, so a training needs profile is drawn up for each staff member before you consider specific courses. Some training can be provided by commercial training companies. Where money is tight, other sources of training should be considered but training should not be abandoned altogether even if it consists only of a team member's being told to find out about a new software tool and then demonstrating it to colleagues. Of course the nice thing about external courses is that one gets to talk to colleagues from other organizations – but attending meetings of your local branch of one of the IS/IT professional associations can serve the same purpose.

The methods learnt need, of course, to be actually applied. Reviews and inspections should help to ensure this.

11.6 Motivation

The third concern that we noted from Taylor was that of motivating people to work. We are now going to look at some different models of motivation that have been proposed.

The Taylorist model

Taylor's viewpoint is reflected in the use of piece-rates in manufacturing industries and sales bonuses amongst sales forces. A problem that project leaders must be aware of is that piece-rates often cause difficulties if a new system is going to change work practices. If new technology is going to improve productivity, the question of adjusting piece-rates downwards to reflect this will be a sensitive issue.

Piece-rates are where workers are paid a fixed sum for each item they produce. Day-rates refer to payment for time worked.

Usually, radical changes in work practices have to be preceded by a move from piece-rates to day-rates.

Group norms are discussed further under group decision making.

Even where work practices are stable and output can be easily related to reward, people paid by the amount they produce will not automatically maximize their output in order to maximize their income. The amount of output will often by constrained by 'group norms', informal, even unspoken, agreements among colleagues about the amount to be produced.

Rewards have to be related in a simple and direct way to the work produced. Where a computer system is being produced, this is not easy. It is difficult to isolate and quantify work done, especially as system development and support is very much a team effort. Typical is the sentiment expressed by one member of staff in a study of software support work practices:

Quoted by Wanda J. Orlikowski in 'Evolving with Notes: Organizational change around groupware technology' in *Groupware & Teamwork*, edited by Claudio U. Ciborra, Wiley and Sons, 1996.

'*This support department does well because we're a team, not because we're all individuals. I think it's the only way the support team can work successfully.*'

In this kind of environment, a reward system that makes excessive distinctions between co-workers can be damaging to morale and eventually to productivity.

Exercise 11.2

A software development department wants to improve productivity by encouraging the re-use of existing software components. It has been suggested that this could be encouraged through financial rewards. To what extent do you think this could be done?

Maslow's hierarchy of needs

Different people are motivated by different things. Clearly money is a very strong motivator when you are broke. However, as the basic need for cash is satisfied, other motivators are likely to emerge. Abraham Maslow, an American psychologist, suggested that there is a hierarchy of needs. As lower levels of needs are satisfied then gradually higher level needs emerge. If these are then satisfied then yet another level of need will emerge. Basic needs are for things like food and shelter. The highest level need, according to Maslow, is the need for 'self-actualization', the feeling that you are completely fulfilling your potential.

In practice, the project leader must realise that people are likely to be motivated by different things at different stages of their life. For example, salary increases, while always welcome, probably have less of an impact on the more mature employee who is already relatively well-paid, than on a new and lowly-paid trainee. Older team-members might place more value on qualities of the job such as being allowed relative autonomy when they do their work, which shows respect for their judgment and sense of responsibility.

Exercise 11.3

Newspapers often report on the vast sums of money that are paid to the top executives of many companies. Does this mean that these people are at a low level

in the Maslow hierarchy of motivation? Do they really need all this money to be motivated? What do you think that the significance of these salaries really is?

Herzberg's two-factor theory

Certain things about a job might make you dissatisfied. If the causes of this dissatisfaction are removed, this does not necessarily make the job more exciting. On the basis of research into job satisfaction that Herzberg and his associates carried out there seemed to be two sets of factors about a job that were of importance:

- **hygiene or maintenance factors**, which can make you dissatisfied if they are not right, for example, the level of pay or the working conditions;
- **motivators**, which make you feel that the job is worthwhile, like a sense of achievement or the nature of the work itself.

Brigette, at Brightmouth College, is in an environment where it is difficult to compete with the high level of maintenance factors that can be provided by a large organization like IOE, but the smaller organization with its closer contact with the users is often able to provide better motivators.

Exercise 11.4

Identify three incidents or times when you felt particularly pleased or happy about something to do with your work or study. Identify three occasions when you were particularly dissatisfied with your work or study. Compare your findings with those of your colleagues and try to identify any patterns.

The expectancy theory of motivation

Amanda and Brigette will need to be aware of how the day-to-day ups and downs of system development affect motivation. A model of motivation developed by Vroom and his colleagues illustrates this. It identifies three influences on motivation:

- **expectancy**, the belief that working harder will lead to a better performance;
- **instrumentality**, the belief that better performance will be rewarded;
- **perceived value**, of the resulting reward.

Motivation will be high when all three factors are high. A zero level for any one of the factors can lead to a lack of motivation.

Imagine that you are trying to get a software package supplied by a third party to work. If you realize that you will never get it to work because of a bug in it, you

will give up. No matter how hard you work you will not be able to do any better (zero expectancy).

If you are working on a package for a user and, although you think you can get it to work, you discover that the user has started employing an alternative package and no longer needs this one, then you will probably feel you are wasting your time and give up (zero instrumentality).

Given that the users really do want the package, your reward in this set of circumstances might simply be a warm feeling that you have helped your colleagues and that they are grateful to you. If in fact, when the users employ the package all they do is complain and hold you responsible for any shortcomings, then you will probably avoid getting involved if they later ask for help implementing a different package (low perceived value of reward).

The Oldham–Hackman job characteristics model

Managers should try to group together the elements of the tasks that need to be carried out so that they form meaningful and satisfying assignments. Oldham and Hackman suggest that the satisfaction that a job gives is based on five factors. The first three factors make the job 'meaningful' to the person who is doing it:

- **skill variety**, the number of different skills that the job holder has the opportunity to exercise;
- **task identity**, the degree to which your work and its results are identifiable as belonging to you;
- **task significance**, the degree to which your job has an influence on others.

The other two factors are:

- **autonomy**, the discretion you have about the way that you do the job;
- **feedback**, the information you get back about the results of your work.

Couger and Zawacki found that programmers in general rated their jobs lower on these factors than other professions, while systems analysts and analyst-programmers rated them higher. Computer development people experienced about the same level of meaningfulness in their work as other, non-IT, professionals, but had lower perceptions of the degree of responsibility and knowledge of results of their work.

Cheney found that in the programming environment, the degree to which programmers got feedback on their work and the degree to which they could contribute to decision making had positive influences on both productivity and job satisfaction, although 'consideration', which was 'the degree to which the leader develops a work climate of psychological support, mutual trust and respect, helpfulness and friendliness', rated as less important.

In practical terms, activities should be designed so that, where possible, staff follow the progress of a particular product and feel personally associated with it.

Methods of improving motivation

- **Setting specific goals** These goals need to be demanding and yet acceptable to staff. Involving staff in the setting of goals helps to gain acceptance for them.
- **Providing feedback** Not only do goals have to be set but staff have to have regular feedback about how they are progressing.
- **Job design** Jobs can be altered to make them more interesting and give staff more feeling of responsibility.

Two measures are often used to enhance job design – job enlargement and job enrichment.

Job enlargement and job enrichment are based on the work of F. Herzberg.

- **Job enlargement** The scope of the job is increased so that the member of staff carries out a wider range of activities. It is the opposite of increasing specialization. For example, a programmer in a maintenance group might be given responsibility for specifying minor amendments as well as carrying out the actual code changes. It is significant that Couger and Zawacki found that programmer/analysts had a higher degree of job satisfaction than programmers.
- **Job enrichment** In this case, the job is changed so that the holder carries out tasks that are normally done at a higher, managerial, level. Staff might be given responsibility for ordering consumables, for scheduling their work or for quality control. With a programmer in a maintenance team, they might be given authority to accept requests for changes which involved less than five days' work without the need for their manager's approval.

11.7 Working in groups

Having discussed people as individuals, we move on to their place in groups. A key problem with major software projects is that they always involve working in groups, but as we have seen many people attracted to computer development find this difficult.

Formal groups can be subdivided into *command groups*, which are the departmental groupings that are seen on organization hierarchy diagrams and which reflect the formal management structure and *task groups* set up to deal with specific tasks. These call on people from different command groups and would typically be disbanded once the task has been completed.

11.8 Becoming a team

Simply throwing people together does not mean that they will immediately be able to work together as a team. Group feelings develop over a period of time. One suggestion is that teams go through five basic stages of development:

This classification is associated with B. W. Tuckman and M. A. Jensen.

- **forming** – the members of the group get to know each other and try to set up some ground rules about behaviour;
- **storming** – conflicts arise as various members of the group try to exert leadership and the group's methods of operation are being established;
- **norming** – conflicts are largely settled and a feeling of group identity emerges;
- **performing** – the emphasis is now on the tasks at hand;
- **adjourning** – the group disbands.

Where people are being put together into a team for the first time, then some specific team-building exercises can be undertaken. Some organizations, for example, send their management teams off on outward bound courses. Without going to these lengths, Amanda and Brigette might try and think of some training activity which could assist in team building.

R. Meredith Belbin *Management Teams: Why They Succeed or Fail*, Heinemann, 1981, contains a self-assessment questionnaire that can help identify to which role a person is best suited.

Valuable research has gone into looking at the best mix of personalities in a project team. Belbin studied teams working together on management games using various mixes of people. He initially tried putting all the people who were most able into one group. Surprisingly, these élite teams tended to do very badly – they argued a lot and as a result important tasks were often neglected.

Belbin came to the conclusion that teams needed a balance of different types of people.

- **The chair** Not necessarily a brilliant leader but must be good at running meetings, being calm, strong but tolerant.
- **The plant** Someone who is essentially very good at generating ideas and potential solutions to problems.

In *Team roles at work*, 1993, Belbin suggests that 'co-ordinator' and 'implementer' are better descriptions than 'chair' and 'team worker'. A new role is added: the 'specialist', the 'techie' who likes to acquire knowledge for its own sake.

- **The monitor-evaluator** Good at evaluating ideas and potential solutions and helping to select the best one.
- **The shaper** Rather a worrier, who helps to direct the team's attention to the important issues.
- **The team worker** Skilled at creating a good working environment, for example by 'jollying people along'.
- **The resource investigator** Adept at finding resources in terms of both physical resources and information.
- **The completer-finisher** Good at completing tasks.
- **The company worker** A good team player who is willing to undertake less attractive tasks if they are needed for team success.

A person can have elements of more than one type. On the other hand, about 30% of the people examined by Belbin could not be classified at all! To be a good team member you must be able to:

- time your interventions, that is, not overwhelm the others in the team;
- be flexible;
- be restrained;
- keep the common goals of the team in mind all the time.

Group performance

Are groups more effective than individuals working alone? Given the preference of many people attracted to software development for working on their own, this is an important question. In many projects, judgements need to be made about which tasks are best carried out collectively and which are best delegated to individuals to do on their own. As one manager at IBM was quoted as saying: '*Some work yields better results if carried out as a team while some things are slowed down if the work is compartmentalized on an individual basis*'. Part of the answer lies in the type of task being undertaken.

The IBM manager was quoted by Angelo Failla in 'Technologies for Co-ordination in a Software Factory' in *Groupware & Teamwork* edited by C. U. Ciborra, Wiley & Sons, 1996.

One way of categorizing group tasks is into:

- additive tasks;
- compensatory tasks;
- disjunctive tasks;
- conjunctive tasks.

Additive tasks are where the efforts of each participant are added together to get the final result, as in a gang of people clearing snow. The people involved are interchangeable.

With *compensatory tasks* the judgements of individual group members are pooled so that errors by some group members are compensated for by the inputs from others. An example of this would be where individual members of a group are asked to provide estimates of the effort needed to produce a piece of software and the results are then averaged. In these circumstances, group work is generally more effective than the efforts of individuals.

Code reviews could be seen as an example of a compensatory task.

With *disjunctive tasks* there is only one correct answer. The effectiveness of the group depends on:

- someone coming up with the right answer;
- the others recognizing it as being correct.

With this type of task, the group can only be as good as its best member and no better.

Conjunctive tasks are where progress is governed by the rate of the slowest performer. Software production where different staff are responsible for different modules seems to be a prime example of this. The overall task is not completed until every participant's work is complete. In this case co-operative attitudes are

productive: the team members who are more advanced need to ensure the meeting of group objectives by assisting those who are behind.

The source of the quotation is the paper by Failla that is cited above.

With all types of collective task, but particularly with additive ones, there is a danger of *social loafing*, where some individuals do not make their proper contribution. This can certainly occur with student group activities, but is not unknown in 'real' work environments. As one software developer has commented: '*[The contribution made to others] is not always recognized. Nor is the lack of any contributions … nobody points out those who fail to make any contributions. Like when there's somebody with vital skills and you ask him for help, but he doesn't provide it*'.

Exercise 11.5

Social loafing is a problem that students often encounter when carrying out group assignments. What steps can participants in a group take to encourage team members to 'pull their weight' properly?

11.9 Decision making

Before we can look more closely at the effectiveness with which groups can make decisions we need to look in general terms at the decision-making process.

Many of the techniques in Chapter 3 are attempts to make decision making more structured.

Decisions can be categorized as being:

- **structured**, generally relatively simple, routine decisions where rules can be applied in a fairly straightforward way;
- **unstructured**, more complex and often requiring a degree of creativity.

Another way of categorizing decisions is by the amount of *risk* and *uncertainty* that is involved.

Many of the techniques in Chapter 3 on project selection are based on the rational-economic model.

Yet another distinction is between the rational-economic model and the satisficing model. The *rational-economic* model of decision making is the basis of classical economics. It predicts, for example, that a prospective buyer of personal computer equipment will purchase goods at the lowest possible price. This assumes that the decision maker has a complete knowledge of the state of the market. In order to achieve this, days, weeks, or months could be spent phoning dealers.

Some research has found that organizations with the most comprehensive solution-seeking techniques are often the poorer financial performers!

Sensible people probably follow a *satisficing* approach and would look at a limited number of representative outlets to get a general idea of prices. Any potential loss of money through having missed an even lower offer would probably be offset by the savings in time, phonecalls, travel and so on.

Some mental obstacles to good decision making

In this book we have rightly stressed a structured, rational, approach to decision making. Many management decisions in the real world, however, made under

pressure and based on incomplete information, are largely intuitive. We have to accept the role of intuition but must be aware that there are some mental obstacles to effective intuitive thinking, for example:

Faulty heuristics Heuristics mean rules of thumb. Rules of thumb can be useful but there are dangers:

- they are based only on the information that is to hand and this can be misleading;
- they are based on stereotypes, such as accepting a Welshman into a male voice choir without an audition because of the 'well-known fact' that the Welsh are a great singing nation.

Escalation of commitment This refers to the way that once you have made a decision it is increasingly difficult to alter it even in the face of evidence that it is wrong.

Information overload It is actually possible to be presented with too much information so that you 'cannot see the wood for the trees'.

Group decision making

There will be occasions where Amanda at IOE, for instance, will want to consult her whole project team about some problem. With a project team, different specialists and points of view can be brought together. Decisions made by the team as a whole are more likely to be accepted than those that are imposed upon it.

A different type of participatory decision-making might occur when end users are consulted about the way a projected computer system is to operate.

Assuming that the meetings are genuinely collectively responsible and have been properly briefed, research shows that groups are better at solving complex problems where the members of the group have complementary skills and expertise. The meeting allows them to communicate freely and to get ideas accepted.

Groups are less effective when dealing with poorly structured problems, which need creative solutions. Brainstorming techniques have been developed to help groups in this situation but research shows that people often come up with more ideas individually than in a group. Where the aim is to get the involvement of end users of a computer system, then prototyping and participatory approaches such as Joint Application Development (JAD) might be adopted.

JAD has been already discussed in Chapter 4.

Obstacles to good group decision making

Amanda finds that group decision making has some disadvantages: it is time consuming; it can in some cases stir up conflicts within the group; and decisions can be unduly influenced by dominant members of the group.

Conflict could, in fact, be less than might be expected. Experiments have shown that people will modify their personal judgements to conform to *group norms*. These are common attitudes that are developed by a group over a period of time.

Once established group norms can survive many changes of membership in the group.

You might think that this would tend to moderate the more extreme views that some individuals in the group might hold. In fact, people in groups often make

decisions that carry more risk than where they have to make the decision on their own. This is known as the *risky shift*.

Measures to reduce the disadvantages of group decision making

One method of making group decision making more efficient and effective is by training members to follow a set procedure. The *Delphi technique* endeavours to collate the judgements of a number of experts without actually bringing them face-to-face. Given a problem, the following procedure is carried out:

- the co-operation of a number of experts is enlisted;
- the problem is presented to the experts;
- the experts record their recommendations;
- these recommendations are collated and reproduced;
- the collected responses are recirculated;
- the experts comment on the ideas of others and modify their recommendations if so moved;
- if the leader detects a consensus then the process is stopped, otherwise the comments are recirculated to the experts.

The big problem with this approach used to be that because the experts could be geographically dispersed the process was time consuming.

Exercise 11.6 What developments in information technology would be of particular assistance to use of the Delphi technique?

11.10 Leadership

When Amanda and Brigette first took on project management responsibilities, one of their private anxieties was a fear that they would not have enough personal authority – that staff would not take them seriously. Leadership is generally taken to mean the ability to influence others in a group to act in a particular way in order to achieve group goals. A leader is not necessarily a good manager or vice versa, because managers have other roles to play, such as those of organizing, planning and controlling.

Authorities on this subject have found it very difficult to agree a list of the common characteristics of good leaders. It would, however, seem safe to say that they seem to have a greater need for power and achievement and have more self-control and more self-confidence than others.

Leadership is based on the idea of some kind of authority or power, although leaders do not necessarily have much formal authority. This power comes from

either the person's position (*position* power) or from the person's individual qualities (*personal* power) or can be a mixture of the two. Position power has been further analysed into:

These ideas are associated with the work of J. R. P. French and B. H. Raven.

- **coercive power**, the ability to force someone to do something by threatening punishment;
- **connection power**, which is based on having access to those who have power;
- **legitimate power**, which is based on a person's title conferring a special status;
- **reward power**, where the holder can confer rewards on those who carry out tasks to their satisfaction.

Personal power, on the other hand, can be further analysed into:

- **expert power**, which comes from being the person who is able to do a specialized task;
- **information power**, where the holder has access to information that others do not;
- **referent power**, which is based on the personal attractiveness of the leader.

Exercise 11.7

What kinds of power (as defined above) would the following people have?

i. An internal auditor looking at the payroll system at Brightmouth College.

ii. A consultant who is called in to advise International Office Equipment about ways of improving software development productivity.

iii. The principal of Brightmouth College who has told staff that they must accept a new contract or face the sack.

iv. Brigette in respect to the users of the college payroll system.

v. Amanda in respect of the people in the project team developing the group maintenance accounts application.

Leadership styles

We have already suggested that Amanda and Brigette were initially concerned about establishing their personal authority. Balanced against this is the need to involve the staff in some of the decision making in order to make the best use of expertise and to gain commitment. Amanda and Brigette will need to judge when they must be authoritative and insist on things and when they must be more flexible and tolerant. Amanda, for example, might decide to be very democratic when formulating plans, but once the plans have been agreed, to insist on a very

disciplined execution of the plan. Brigette, on the other hand, might find at Brightmouth College that she alone has the technical expertise to make some decisions, but, once she has briefed people on what needs to be done, they expect to be left alone to get on with the job as they best see fit.

Attempts have been made to measure leadership styles on two axes: directive vs. permissive and autocratic vs. democratic:

This approach is associated with Rensis Likert.

- **directive autocrat** makes decisions alone with close supervision of their implementation;
- **permissive autocrat** makes decision alone but gives subordinates latitude in implementation;
- **directive democrat** makes decisions participatively but closely supervises their implementation;
- **permissive democrat** makes decisions participatively and gives subordinates latitude in implementation.

It should be emphasized that there is no one best style of management – it depends on the situation.

Another axis on which there have been attempts to measure management qualities has been on the degree to which a manager is *task-oriented*, that is, the extent to which the execution of the task at hand is paramount, and the degree to which the manager is concerned about the people involved (*people orientation*). It is perhaps not surprising that subordinates appear to perform best with managers who score highly in both respects.

Work environments vary according to the amount of control that can be exerted over the work. Some jobs are routine and predictable (as when dealing with batched computer output). Others may be driven by outside factors (as in the case of a help-desk) or are situations where future direction is uncertain (for example, at the early stages of a feasibility study). Where there is a high degree of uncertainty, subordinates will seek guidance from above and welcome a task-oriented management style. As uncertainty is reduced, the task-oriented manager is likely to relax and to become more people-oriented and this will have good results. People-oriented managers are better where staff can control the work they do and know what to do without referring matters to their line managers. It is then argued that if control becomes even easier the people-oriented manager will be tempted to get involved in more task-centred questions and that this may have undesirable results.

Research findings also show that where team members are relatively inexperienced a task-oriented approach is most effective. As group members mature, consideration for their personal needs and aspirations becomes more valued. Where maturity is very high, then there is no need for a strong emphasis on either of these approaches.

Exercise 11.8

What in your view would be the most appropriate management style when dealing with the following subordinates?

i. At Brightmouth College, a former member of the local authority who has dealt with the college payroll for several years and who has been employed by the college to set up and manage the new payroll section.

ii. At IOE, a new trainee analyst–programmer who has just joined Amanda's group.

iii. At IOE, a very experienced analyst–programmer aged 45, who was recruited into the software development department some time ago from the accounts department and who has been dealing with system support for the old maintenance accounts system that is now being revised.

11.11 Organizational structures

Formal versus informal structures

While organizational structures can have an enormous impact on the way a project is conducted, it is something that project leaders such as Amanda at IOE can often do little to change.

The *formal* structure is the one that is expressed in the staff hierarchy chart. It is basically concerned with *authority*, about who has which boss. It is backed by an *informal* structure of contacts and communication that grows up spontaneously among members of staff during the course of work. When the unexpected happens it is often this system that comes into play. Over a period of time, the advantages and disadvantages of different organizational structures tend to even out – the informal organization gets built up and staff find unofficial ways of getting around the obstacles posed by the formal structure.

Hierarchical approach

The 'traditional' management structure is based on the concept of the *hierarchy* – each member of staff has only one manager, while a manager will have responsibility for several members of staff. Authority flows from the top down through the structure. A traditional concern has been with the *span of control* – the number of people that a manager can effectively control.

Staff versus line

Staff in organizations can often be divided into *line* workers who actually produce the end product and support *staff* who carry out supporting roles. In some organizations that produce software for the market or as a component of a larger product which is sold, the software specialists might be seen as part of the line. In a financial organization, on the other hand, the information systems department would probably be seen as part of the support staff.

Departmentalization

In drawing up a structure, the question of *differentiation* crops up. This is the question of how the organization is to be departmentalized. This is often based on staff specialisms, product lines, categories of customer or geographical location, for example.

In the case of software development, it is usually the case that either a *functional* or a *task-oriented* approach is used. With functional departmentalization, systems analysts might be put in a group separate from the programmers. The programmers would act as a pool from which resources may be drawn for particular tasks. With a task-oriented approach, the programmers and systems analysts are grouped together in one project team. The project team might be gathered in order to implement a specific long-term project or might exist on a permanent basis to service the needs of a particular set of users.

One advantage of the functional approach is that it can lead to a more effective use of staff. Programmers can be allocated to jobs as needed and be released for other work when a particular task is completed. For instance, in a project team there are bound to be periods of greater and lesser coding activity and programmers might find there are spells when they are under-utilized. The functional organization will also make it easier for programmers to have careers that are technically oriented – there will probably be a career structure within the software development department that allows the programmer to rise without having to change specialism. This type of organization should also encourage the interchange of new technical ideas among technical staff and the promulgation of company wide standards.

A disadvantage is that having two separate departments can lead to communication problems, especially if a programmer is unfamiliar with the application area. There will also be problems with software maintenance – here it is helpful to have programmers who have built up a familiarity with particular parts of the application software. Users might prefer the established project team approach because, when they require new software features, they will already have a group dedicated to their needs and will not find themselves in the position of always having to fight other departments for development resources. The project team structure tends to favour a pattern of career progression where programmers eventually become systems analysts.

A third method of departmentalization is based on lifecycle phase. Here there are separate teams for development and maintenance. Some staff can concentrate in a focused and sustained manner on developing new applications with few interruptions, while other teams, more oriented towards service and support, deal with maintenance.

Some organizations have attempted to get the best of all worlds by having a *matrix* structure. In this case the programmer would have two managers: a project leader who would give day-to-day direction about the work in hand and a programming manager who would be concerned about such things as career development.

Centralized versus decentralized group structures

At the level of a project group, a decentralized organization would mean that the group members would tend to make major decisions collectively and that there would be a large degree of free communication among group members. With the centralized approach the group would be broken down into sections, each of which would be directed by a leader who communicates on behalf of the section with other groups.

Decentralized groups, because of the time taken to debate things, tend to work more slowly. They are likely to be affected by the establishment of group norms and the influence of the risky shift, which has already been described. However, they are better at dealing with complex problems while the centralized group organization deals more effectively with simple problems.

The discussion of centralized versus decentralized groups assumes that software development work has to be done as a group. In fact, given the preference of many software developers for working on their own, an organization where each programmer works in isolation can be envisaged – indeed there are software houses that are based on people working at home.

Egoless programming

In the early days of computer development, managers tended to think of the programmer as communing mysteriously with the machine. The tendency was for programmers to see programs as being an extension of themselves and to feel over-protective towards them. The effects of this on the maintainability of programs can be imagined. Gerald Weinberg made the then revolutionary suggestion that programmers and programming team leaders should read other people's programs. Programs would become in effect the common property of the programming group and programming would become 'egoless'. Peer code reviews are based on this idea. Weinberg's ideal programming team was a decentralized group freely communicating within itself.

G. M. Weinberg, *The Psychology of Computer Programming*, Van Nostrand Reibold, 1971.

Chief programmer teams

The larger the decentralized group, the slower it will get, because of the increased communication. On really large time-critical projects, a more formalized centralized structure is essential. Brooks pointed out the need for design consistency when producing a large complex system and how this might be difficult when there are a large number of people involved in producing a piece of software. One suggestion was to try to reduce the number of people actually creating software but to make these programmers as productive as possible by giving them as much support as possible.

The result of this train of thought was the *chief programmer* team. The chief programmer is the person who defines the specification, and designs, codes, tests and documents the software. There is also a *copilot*, with whom the chief programmer can discuss problems and who writes some code. They are supported by an *editor* to write up the documentation drafted by the chief programmer, a

Brooks' *Mythical Man-Month* has already been referred to. He was in charge of the huge team that created the operating system for the IBM 360 range.

program clerk to maintain the actual code, and a *tester*. The general idea is that this team is under the control of a single unifying intellect.

The chief programmer concept was used on the influential *New York Times* data bank project, where many aspects of structured programming were tried out. In this case, each chief programmer managed a senior level programmer and a program librarian. Additional members could be added to the team on a temporary basis to deal with particular problems or tasks.

The problem with this kind of organization is getting hold of really outstanding programmers to carry out the chief programmer role. There are also the dangers of information overload on the chief programmer, and of staff dissatisfaction among those who are there simply to minister to the needs of the superstar chief programmers.

Controlled decentralized groups

This compromise structure has been suggested and seems to follow common industry practice. A project team is made of groups under the leadership of senior programmers. Within these groups there is free communication and a practice of reviewing each others' work. Communication with other groups is at senior programmer level, while a project leader has overall authority.

11.12 Conclusion

Some of the important points that have been made in this chapter are:

- people may be motivated by money, but they are motivated by other things as well;
- both staff selection and the identification of training needs should be done in an orderly, structured, way where requirements are clearly defined first;
- thoughtful job design can increase staff motivation;
- consideration should be given, when forming a new project team, to getting the right mix of people and to planning activities that will promote team building;
- group working is more effective with some types of activity than others;
- different styles of leadership are needed in different situations;
- the people who need to communicate most with each other should be grouped together organizationally.

11.13 Further exercises

1. An organization has detected low job satisfaction in the following departments:

- the system testing group;

- the computer applications help desk;
- computer batch input.

How could these jobs be redesigned to give more job satisfaction?

2. In Exercise 11.1, a job specification was requested.
   (a) Write a job holder profile of the sort of person who would be able to fulfil the specification in terms of qualities, qualifications, previous education and experience.
   (b) For each element in the job holder profile that you have produced in (a) above, describe ways of finding out whether an applicant has met the requirement.
3. To what extent is the Belbin approach to balanced teams compatible with having chief programmer teams?
4. If you have been involved recently in a group activity or project, try and categorize each participant according to the Belbin classification. Were there any duplications or gaps in any of the roles? Did this seem to have any impact on progress?
5. Three different mental obstacles to good decision making were identified in the text: faulty heuristics, escalation of commitment and information overload. What steps do you think can be taken to reduce the danger of each of these?
6. In Exercise 11.8, the management style most appropriate for each of three different situations was asked for. Go back and consider how you as a manager would respond to each of these three situations in terms of practical things to do or avoid.