Chapter 1

Introduction to software project management

OBJECTIVES

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

- define the scope of 'software project management';
- distinguish between software and other types of development project;
- understand some problems and concerns of software project managers;
- define the usual stages of a software project;
- explain the main elements of the role of management;
- understand the need for careful planning, monitoring and control;
- identify the stakeholders of a project, their objectives and ways of measuring the success in meeting those objectives;
- measure the success of a project in meeting its objectives.

1.1 Introduction

What exactly do we mean by 'software project management'? To answer this, we need to look at some key ideas about the planning, monitoring and control of software projects. Projects to produce software are worthwhile only if they satisfy real needs and so we will examine how we can identify the stakeholders in a project and their objectives. Identifying those objectives and checking that they are met is the basis of a successful project. This, however, cannot be done unless there is accurate information and how this is provided will be explored.

1.2 What is a project?

Dictionary definitions of 'project' include:
'A specific plan or design'
'A planned undertaking'
'A large undertaking: for example, a public works scheme'
Longman Concise English Dictionary, 1982.

The dictionary definitions put a clear emphasis on the project's being a planned activity.

Another key aspect of a project is that the undertaking is non-routine: a job which is repeated a number of times is not a project. There is a hazy boundary in between. The first time you do a routine task it will be very like a project. On the other hand, a project to develop a system that is very similar to previous ones that you have developed will have a large element of the routine.

We can summarize the key characteristics that distinguish projects as follows:

- non-routine tasks are involved;
- planning is required;
- specific objectives are to be met or a specified product is to be created;
- the project has a predetermined time span (which may be absolute or relative);
- work is carried out for someone other than yourself;
- work involves several specialisms;
- work is carried out in several phases;
- the resources that are available for use on the project are constrained;
- the project is large or complex.

In general, the more any of these factors applies to a task, the more difficult it is going to be to complete it successfully. Project size is particularly important. It should not be a surprise that a project that employs 200 project personnel is going to be rather trickier to manage than one that involves just two people. The examples and exercises used in this book usually relate to smaller projects. This is just to make them more manageable from a learning point of view: the techniques and issues discussed are of equal relevance to larger projects.

Exercise 1.1

Consider the following:

- producing an edition of a newspaper;
- building the Channel Tunnel;
- getting married;
- amending a financial computer system to deal with dates after the 31st December, 1999;
- a research project into what makes a good human–computer interface;
- an investigation into the reason why a user has a problem with a computer system;

- a programming assignment for a second year computing student;
- writing an operating system for a new computer;
- installing a new version of a word processing application in an organization.

Some would appear to merit the description 'project' more than others. Put them into an order that most closely matches your ideas of what constitutes a project. For each entry in the ordered list, describe the difference between it and the one above that makes it less worthy of the term 'project'.

There is no one correct answer to this exercise, but a possible solution to this and the other exercises you will come across may be found at the end of the book.

1.3 Software projects versus other types of project

Many of the techniques of general project management are applicable to software project management, but Fred Brooks pointed out that the products of software projects have certain characteristics that make them different.

Brooks, F. P. 'No silver bullet: essence and accidents of software engineering'. This essay has been included in *The Mythical Man-Month*, Anniversary Edition, Addison-Wesley, 1995.

One way of perceiving software project management is as the process of making visible that which is invisible.

Invisibility When a physical artefact such as a bridge or road is being constructed the progress being made can actually be seen. With software, progress is not immediately visible.

Complexity Per dollar, pound or euro spent, software products contain more complexity than other engineered artefacts.

Flexibility The ease with which software can be changed is usually seen as one of its strengths. However this means that where the software system interfaces with a physical or organizational system, it is expected that, where necessary, the software will change to accommodate the other components rather than vice versa. This means the software systems are likely to be subject to a high degree of change.

1.4 Activities covered by software project management

A software project is concerned not only with the actual writing of software. In fact, where a software application is bought in 'off-the-shelf', there might be no software writing as such. This is still fundamentally a software project because so many of the other elements associated with this type of project are present.

Chapter 4 on project analysis and technical planning looks at some alternative life cycles.

Usually, there are three successive processes that bring a new system into being:

1. **The feasibility study** This is an investigation to decide whether a prospective project is worth starting. Information will be gathered about the general requirements of the proposed system. The probable developmental

and operational costs, along with the value of the benefits of the new system are estimated. With a large system, the feasibility study could be treated as a project in its own right. This evaluation may be done as part of a strategic planning exercise where a whole range of potential software developments are evaluated and put into an order of priority. Sometimes an organization has a policy where a series of projects is planned as a *programme* of development.

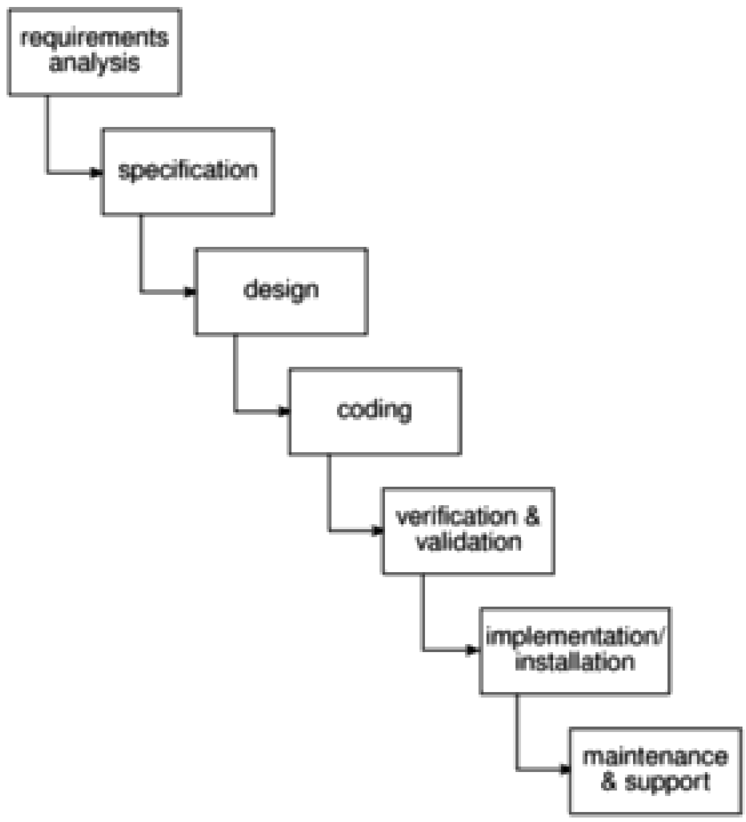

Figure 1.1 *A typical project life-cycle.*

The PRINCE 2 method, which is described in Appendix A takes this planning by stages approach.

2. **Planning** If the feasibility study produces results that indicate that the prospective project appears viable, then planning of the project can take place. In fact, for a large project, we would not do all our detailed planning right at the beginning. We would formulate an outline plan for the whole project and a detailed one for the first stage. More detailed planning of the later stages would be done as they approached. This is because we would have more detailed and accurate information upon which to base our plans nearer to the start of the later stages.

3. **Project execution** The project can now be executed. Individual projects are likely to differ considerably but a classic project life-cycle is shown in Figure 1.1.

The stages in the life-cycle illustrated in Figure 1.1 above are described in a little more detail below:

Requirements analysis This is finding out in detail what the users require of the system that the project is to implement. Some work along these lines will almost certainly have been carried out when the project was evaluated but now the original information obtained needs to be updated and supplemented. Several different approaches to the users' requirements may be explored. For example, a small system that satisfies some, but not all, of the users' needs at a low price may be compared to a system with more functions but at a higher price.

Specification Detailed documentation of what the proposed system is to do.

Design A design that meets the specification has to be drawn up. This design activity will be in two stages. One will be the external or user design. This lays down what the system is to look like to the users in terms of menus, screen and report layouts and so on. The next stage produces the physical design, which tackles the way in which the data and software procedures are be structured internally.

Coding This might refer to writing code in a procedural language such as C or Ada, or might refer to the use of a high level application builder. Even where software is not being built from scratch, some modification to the base application might be required to meet the needs of the new application.

Verification and validation Whether software is developed specially for the current application or not, careful testing will be needed to check that the proposed system meets its requirements.

Implementation/installation Some system development practitioners refer to the whole of the project after design as 'implementation' (that is, the implementation of the design) while others insist that the term refers to the installation of the system after the software has been developed. In this case it encompasses such things as setting up data files and system parameters, writing user manuals and training users of the new system.

Maintenance and support Once the system has been implemented there will be a continuing need for the correction of any errors that may have crept into the system and for extensions and improvements to the system. Maintenance and support activities may be seen as a series of minor software projects. In many environments, most software development is in fact maintenance.

Exercise 1.2

Brightmouth College is a higher education institution which used to be managed by the local government authority but has now become autonomous. Its payroll is still administered by the local authority and pay slips and other output are produced in the local authority's computer centre. The authority now charges the college for this service. The college management are of the opinion that it would be cheaper to obtain an 'off-the-shelf' payroll application and do the payroll processing themselves.

What would be the main stages of the project to convert to independent payroll processing by the college? Bearing in mind that an off-the-shelf application is to be used, how would this project differ from one where the software was to be written from scratch?

1.5 Some ways of categorizing software projects

It is important to distinguish between the main types of software project because what is appropriate in one context might not be so in another. For example, SSADM, the Structured Systems Analysis and Design Method, is suitable for developing information systems but not necessarily other types of system.

Information systems versus embedded systems

Embedded systems are also called real-time or industrial systems.

A distinction may be made between *information systems* and *embedded systems*. Very crudely, the difference is that in the former case the system interfaces with the organization, whereas in the latter case the system interfaces with a machine! A stock control system would be an information system that controls when the organization reorders stock. An embedded, or process control, system might control the air conditioning equipment in a building. Some systems may have elements of both so that the stock control system might also control an automated warehouse.

Exercise 1.3

Would an operating system on a computer be an information system or an embedded system?

Objectives versus products

Projects may be distinguished by whether their aim is to produce a *product* or to meet certain *objectives*.

A project might be to create a product the details of which have been specified by the client. The client has the responsibility for justifying the product.

Service level agreements are becoming increasingly important as organizations contract out functions to external service suppliers.

On the other hand, the project might be required to meet certain objectives. There might be several ways of achieving these objectives in contrast to the constraints of the product-driven project. One example of this is where a new information system is implemented to improve some service to users inside or outside an organization. The subject of an agreement would be the level of service rather than the characteristics of a particular information system.

Many software projects have two stages. The first stage is an objectives-driven project, which results in a recommended course of action and may even specify a new software application to meet identified requirements. The next stage is a project actually to create the software product.

Exercise 1.4

Would the project to implement an independent payroll system at the Brightmouth College described in Exercise 1.2 above be an objectives-driven project or a product-driven project?

1.6 The project as a system

A project is concerned with creating a new system and/or transforming an old one and is itself a system.

Systems, subsystems and environments

A simple definition of the term *system* is 'a set of interrelated parts'. A system will normally be part of a larger system and will itself comprise *subsystems*.

Outside the system there will be the system's *environment*. This will be made up of things that can affect the system but over which the system has no direct control. In the case of Brightmouth College, the bankruptcy of the main supplier of IT equipment would be an event happening in the system's environment.

Exercise 1.5

Identify the possible subsystems of the installed Brightmouth College payroll system.

What important entities exist in the payroll system's environment?

Open versus closed systems

Open systems are those that interact with the environment. Nearly all systems are open. One reason that engineered systems and the projects to construct them often fail is that the technical staff involved do not appreciate the extent to which systems are open and are liable to be affected by outside changes.

Sub-optimization

This is where a subsystem is working at its optimum but is having a detrimental effect on the overall system. An example of this might be where software developers deliver to the users a system that is very efficient in its use of machine resources, but is also very difficult to modify.

Sociotechnical systems

Software projects belong to this category of systems. Any software project requires both technological organization and also the organization of people. Software project managers therefore need to have both technical competence and the ability to interact persuasively with other people.

1.7 What is management?

The Open University suggest that management involves the following activities:

A convenient way of accessing this OU material is in D. Ince, H. Sharp, and M. Woodman, *Introduction to Software Project Management and Quality Assurance*, McGraw-Hill, 1993.

- planning – deciding what is to be done;
- organizing – making arrangements;
- staffing – selecting the right people for the job, for example;
- directing – giving instructions;
- monitoring – checking on progress;
- controlling – taking action to remedy hold-ups;
- innovating – coming up with new solutions;
- representing – liaising with users etc.

Exercise 1.6

Paul Duggan is the manager of a software development section. On Tuesday at 10.00 am he and his fellow section heads have a meeting with their group manger about the staffing requirements for the coming year. Paul has already drafted a document 'bidding' for staff. This is based on the work planned for his section for the next year. The document is discussed at the meeting. At 2.00 pm Paul has a meeting with his senior staff about an important project his section is undertaking. One of the software development staff has just had a road accident and will be in hospital for some time. It is decided that the project can be kept on schedule by transferring another team member from less urgent work to this project. A temporary replacement is to be brought in to do the less urgent work but this might take a week or so to arrange. Paul has to phone both the personnel manager about getting a replacement and the user for whom the less urgent work is being done explaining why it is likely to be delayed.

Identify which of the eight management responsibilities listed above Paul was responding to at different points during his day.

Another way of looking at the management task is to ask managers what their most frequent challenges are. A survey of software project managers produced the following list:

The results of this survey by H. J. Thamhain and D. L. Wilemon appeared in June 1986 in *Project Management Journal* under the title 'Criteria for controlling software according to plan'.

- coping with deadlines (85%);
- coping with resource constraints (83%);
- communicating effectively among task groups (80%);
- gaining commitment from team members (74%);
- establishing measurable milestones (70%);
- coping with changes (60%);

- working out project plan agreement with their team (57%);
- gaining commitment from management (45%);
- dealing with conflict (42%);
- managing vendors and sub-contractors (38%).

Similar lists appear in the computer trade press, for example in the 27 August 1998 edition of *Computing*.

The percentages relate to the numbers of managers identifying each challenge. A manager could identify more than one.

1.8 Problems with software projects

One way of deciding what ought to be covered in 'software project management' is to consider what problems need to be addressed.

Traditionally, management has been seen as the preserve of a distinct class within the organization. As technology has made the tasks undertaken by an organization more sophisticated, many management tasks seem to have become dispersed throughout the organization: there are management systems rather than managers. Nevertheless, the successful project will normally have one person who is responsible for its success. Such people are likely to be concerned with the key areas that are most likely to prevent success – they are primarily trouble-shooters and their job is likely to be moulded by the problems that confront the project. A survey of managers published by Thayer, Pyster and Wood identified the following commonly experienced problems:

- poor estimates and plans;
- lack of quality standards and measures;
- lack of guidance about making organizational decisions;
- lack of techniques to make progress visible;
- poor role definition – who does what?
- incorrect success criteria.

Further details of the survey can be found in 'Major issues in software engineering project management' in *IEEE Transactions on Software Engineering*, Volume 7, pp 333–342.

The above list looks at the project from the manager's point of view. What about the staff who make up the members of the project team? Below is a list of the problems identified by a number of students on a degree course in Computing and Information Systems who had just completed a year's industrial placement:

- inadequate specification of work;
- management ignorance of IT;
- lack of knowledge of application area;
- lack of standards;
- lack of up-to-date documentation;

- preceding activities not completed on time – including late delivery of equipment;
- lack of communication between users and technicians;
- lack of communication leading to duplication of work;
- lack of commitment – especially when a project is tied to one person who then moves;
- narrow scope of technical expertise;
- changing statutory requirements;
- changing software environment;
- deadline pressure;
- lack of quality control;
- remote management;
- lack of training.

Note how many of the problems identified by the students stemmed from poor communications. Another common problem identified by this and other groups of students is the wide range of IT specialisms – an organization may be made up of lots of individuals or groups who will be expert in one set of software techniques and tools but ignorant of those used by their colleagues. Communication problems are therefore bound to arise.

Stephen Flower's *Software Failure, Management Failure*, Wiley & Sons, 1996, is an interesting survey of failed computer projects

What about the problems faced by the customers of the products of computer projects? Here are some recent stories in the press:

- the United States Internal Revenue System was to abandon its tax system modernization programme after having spent $4 billion;
- the state of California spent $1 billion on its non-functional welfare database system;
- the £339 million United Kingdom air traffic control system was reported as being two years behind schedule;
- a discount stock brokerage company had 50 people working 14 hours or more a day to correct three months of records clerically—the report commented that the new system had been rushed into operation without adequate testing;
- in the United Kingdom, a Home Office immigration service computerization project was reported as having missed two deadlines and was nine months late;
- the Public Accounts Committee of the House of Commons in the United Kingdom blamed software bugs and management errors for £12 million of project costs in relation to an implementation of a Ministry of Agriculture computer system to administer farm subsidies.

Most of the stories above relate to public sector organizations. This may be misleading—private sector organizations tend to conceal their disasters and in any case many of the public projects above were actually being carried out by private sector contractors. Any lingering faith by users in the innate ability of IT people to plan ahead properly will have been removed by consideration of the 'millennium bug', a purely self-inflicted IT problem. On balance it might be a good idea not to survey users about their problems with IT projects!

1.9 Management control

The project control cycle

Management, in general, can be seen as the process of setting objectives for a system and then monitoring the system to see what its true performance is. In Figure 1.2 the 'real world' is shown as being rather formless. Especially in the case of large undertakings, there will be a lot going on about which management should be aware. As an example, take an IT project that is to replace locally held paper-based records with a centrally-organized database. It might be that staff in a large number of offices that are geographically dispersed need training and then need to use the new IT system to set up the back-log of manual records on the new database. It might be that the system cannot be properly operational until the last record has been transferred. It might also be the case that the new system will be successful only if new transactions can be processed within certain time cycles. The managers of the project ought to be asking questions about such things as how effective training has been, how many records have still to be transferred to the new database and transfer rates. This will involve the local managers in *data collection*. Bare details, such as 'location X has processed 2000 documents' will not be very useful to higher management: *data processing* will be needed to transform this raw *data* into useful *information*. This might be in such forms as 'percentage of records processed', 'average documents processed per day per person' and 'estimated completion date'.

In the example above, the project leader might examine the 'estimated completion date' for completing data transfer for each branch and compare this with the overall target date for completion of this phase of the project. In effect they are comparing actual performance with one aspect of the overall project objectives. They might find that one or two branches are not going to complete the transfer of details in time, and would then need to consider what to do (this is represented in Figure 1.2 by the box *making decisions/plans*). One possibility would be to move staff temporarily from one branch to another. If this is done, there is always the danger that while the completion date for the one branch is pulled back to before the overall target date, the date for the branch from which staff are being moved is pushed forward beyond that date. The project manager would need to calculate carefully what the impact would be in moving staff from particular branches. This is *modelling* the consequences of a potential solution.

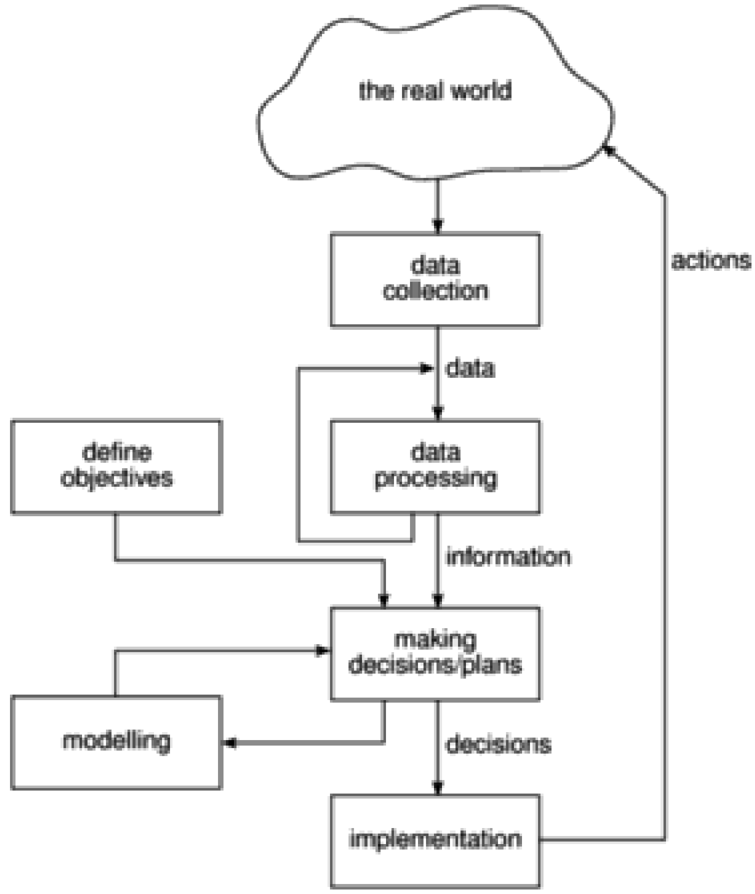

Figure 1.2 *The project control cycle.*

Several different proposals could be modelled in this way before one was chosen for *implementation*.

Having implemented the decision, the situation needs to be kept under review by collecting and processing further progress details. For instance, the next time that progress is reported, a branch to which staff have been transferred might still be behind in transferring details. This might be because the reason why the branch has got behind in transferring details is because the manual records are incomplete and another department, for whom the project has a low priority, has to be involved in providing the missing information. In this case, transferring extra staff to do data input will not have accelerated data transfer.

Objectives

Project objectives should be clearly defined.

To have a successful software project, the manager and the project team members must know what will constitute success. This will make them concentrate on what is essential to project success.

There might be more than one set of users of a system and there might be different groups of staff who are involved its development. There is a need for well defined objectives that are accepted by all these people. Where there is more than one user group, then a *project authority* needs to be identified. Such a project authority has overall authority over what the project is to achieve.

This authority is often held by a *project steering committee*, which has overall responsibility for setting, monitoring and modifying objectives. The project manager still has responsibility for running the project on a day to day basis, but has to report to the steering committee at regular intervals. Only the steering committee can authorize changes to the project objectives and resources.

This committee is likely to contain user, development and management representatives.

Measures of effectiveness

Effective objectives are concrete and well defined. Vague aspirations such as 'to improve customer relations' are unsatisfactory. Objectives should be such that it is obvious to all whether the project has been successful or not. Ideally there should be *measures of effectiveness*, which tell us how successful the project has been. For example, 'to reduce customer complaints by 50%' would be more satisfactory as an objective than 'to improve customer relations'.

The measure can, in some cases, be an answer to simple yes/no question, 'Did we install the new software by 1st June?' for example.

Sub-objectives and goals

In order to keep things manageable, objectives might need to be broken down into sub-objectives. Here we say that in order to achieve A we must achieve B, C and D first. These sub-objectives are also known as *goals*, steps on the way to achieving an objective, just as goals scored in a football match are steps towards the objective of winning the match.

Exercise 1.7

Identify the objectives and sub-objectives of the Brightmouth College payroll project. What measures of effectiveness could be used to check the success in achieving the objectives of the project?

1.10 Stakeholders

These are people who have a stake or interest in the project. It is important that they be identified as early as possible, because you need to set up adequate communication channels with them right from the start. The project leader also has to be aware that not everybody who is involved with a project has the same motivation and objectives. The end users might, for instance, be concerned about the ease of use of the system while their managers might be interested in the staff savings the new system will allow.

Stakeholders might be internal to the project team, external to the project team but in the same organization, or totally external to the organization.

- **Internal to the project team** This means that they will be under the direct managerial control of the project leader.
- **External to the project team but within the same organization** For example, the project leader might need the assistance of the information

management group in order to add some additional data types to a database or the assistance of the users to carry out systems testing. Here the commitment of the people involved has to be negotiated.

- **External to both the project team and the organization** External stakeholders might be customers (or users) who will benefit from the system that the project implements or contractors who will carry out work for the project. One feature of the relationship with these people is that it is likely to be based on a legally binding contract.

Within each of the general categories there will be various groups. For example, there will be different types of user with different types of interests.

B. W. Boehm and R. Ross 'Theory W Software Project Management: Principles and Examples', in B. W. Boehm (ed.) *Software Risk Management.*

Different types of stakeholder might have different objectives and one of the jobs of the successful project leader is to recognize these different interests and to be able to reconcile them. It should therefore come as no surprise that the project leader needs to be a good communicator and negotiator. Boehm and Ross proposed a 'Theory W' of software project management where the manager concentrates on creating situations where all parties involved in a project benefit from it and therefore have an interest in its success. (The 'W' stands for Everyone a Winner.)

Exercise 1.8

Identify the stakeholders in the Brightmouth College payroll project.

1.11 Requirement specification

Very often, especially in the case of product-driven projects, the objectives of the project are carefully defined in terms of functional requirements, quality requirements, and resource requirements.

- **Functional requirements** These define what the system that will be the end product of the project is to do. Systems analysis and design methods, such as SADT and Information Engineering, are designed primarily to provide functional requirements.

These are sometimes called non-functional requirements.

- **Quality requirements** There will be other attributes of the system to be implemented that do not relate so much to what the system is to do but how it is to do it. These are still things that the user will be able to experience. They include, for example, response time, the ease of using the system and its reliability.

- **Resource requirements** A record of how much the organization is willing to spend on the system. There will usually be a trade-off between this and the time it takes to implement the system. In general it costs disproportionately more to implement a system by an earlier date than a later one. There might also be a trade-off between the functional and quality requirements and cost. We would all like exceptionally reliable and user-friendly systems that do exactly what we want but we might not be able to afford them.

1.12 Information and control in organizations

Hierarchical information and control systems

With small projects, the project leaders are likely to be working very closely with the other team members and might even be carrying out many non-managerial tasks themselves. Therefore they should have a pretty good idea of what is going on. When projects are larger, many separate teams will be working on different aspects of the project and the overall managers of the project are not going to have day-to-day direct contact with all aspects of the work.

The larger the project, the bigger the communication problems.

Larger projects are likely to have a *hierarchical* management structure (Figure 1.3). Project team members will each have a group leader who allocates them work and to whom they report progress. In turn the group leader, along with several other group leaders, will report to a manager at the next higher level. That manager might have to report to another manager at a higher level, and so on.

There might be problems that cannot be resolved at a particular level. For example, additional resources might be needed for some task, or there might be a disagreement with another group. These will be referred to the next higher level of management.

The referral of disagreements to a higher level is sometimes known as *escalation*.

At each higher level more information will be received by fewer people. There is thus a very real danger that managers at the higher levels might be overloaded with too much information. To avoid this, at each level the information will have to be summarized.

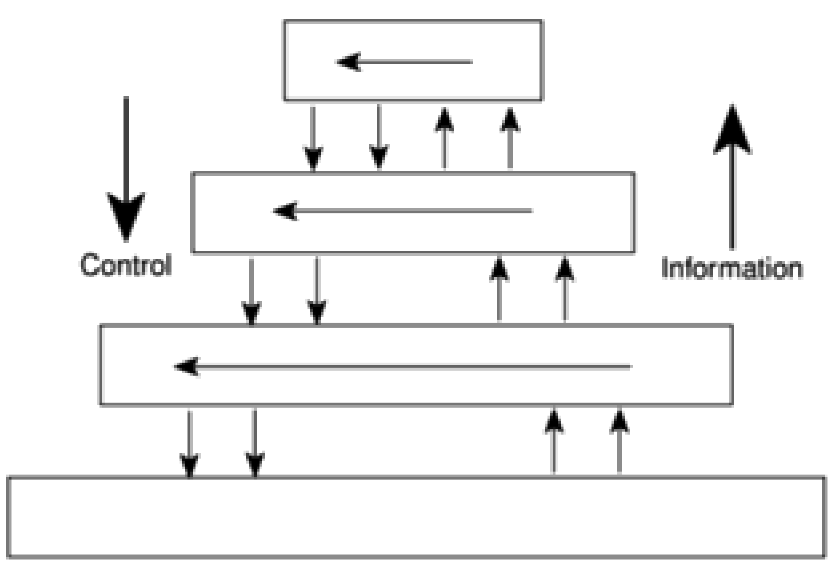

Figure 1.3 *Management information flows up the organizational structure and control flows down.*

As a result of examining the progress information and comparing it against what was planned, some remedial action might need to be taken. Instructions may be formulated and passed down to a lower level of management. The lower level managers will have to interpret what needs to be done and formulate more detailed plans to fulfil the directive. As the directives filter down the hierarchy, they will be expanded into more detail at each level.

For example, a programmer will receive a specification from an analyst and might then seek clarification.

Not all information flows concerning a project will be going up and down the hierarchy. There will also be lateral flows between groups and individuals on the same level.

Levels of decision making and information

Each decision made in a project environment should be based on adequate information of the correct sort. The type of information needed depends on the level of decision making. Decisions can be grouped at three levels: *strategic*, *tactical*, and *operational*.

Strategic decision making is essentially about deciding objectives. In the case of the Brightmouth College payroll, the decision to become administratively independent could be regarded as a strategic decision. In our example we were interested only in the payroll, but this might have been part of a wider programme which may have affected many other administrative functions.

Tactical decision making is needed to ensure that the objectives will be fulfilled. The project leader who has the responsibility for achieving objectives will have to formulate a plan of action to meet those objectives. The project leader will need to monitor progress to see whether these objectives are likely to be met and to take action where needed to ensure that the things remain on course.

Operational decisions relate to the day-to-day work of implementing the project. Deciding the content of the acceptance tests might come under this heading.

Differences in types of information

Table 1.1 gives some idea of the differences in the kind of information needed. There is a kind of continuum for most of the qualities suggested and what is needed for tactical decision making comes somewhere in the middle.

Effectiveness is concerned with doing the right thing. *Efficiency* is carrying out a task making the best possible use of the resources.

Measurement

The quantification of measures of effectiveness reduces ambiguity.

The leader of a small project will have direct contact with many aspects of the project. With larger projects, project leaders would have to depend on information being supplied to them. This information should not be vague and ideally should be quantitative. This ties in with our need for unambiguous measures of effectiveness.

Software development deals largely with intangibles and does not easily lend itself to quantitative measures, but attempts are increasingly being made to introduce measurement into the software process.

Table 1.1 *The types of information required for decision making*

Characteristic	*Operational*	*Strategic*
motivation	efficiency	effectiveness
orientation	internal	internal and external
focus	specific to a function	specific to organization
detail	detailed	summarized
response	fast	not so fast
access paths	standard	flexible
up-to-dateness	essential	desirable
accuracy	essential	approximate
certainty	essential	often predictive
objectivity	high	more subjective
information type	mainly quantitative	often qualitative

Software measurements can be divided into *performance measures* and *predictive measures*.

- **Performance measures** These measure the characteristics of a system that has been delivered. They are important when we are trying to specify unambiguously the quality requirements of a proposed system.
- **Predictive measures** The trouble with performance measures is that you need to have a system actually up and running before you can take measurements. As a project leader, what you want to be able to do is to get some idea of the likely characteristics of the final system during its development. *Predictive measures* are taken during development and indicate what the performance of the final system is likely to be.

Performance measures include mean time between failures (reliability) and time to learn an application (usability).

For example, the errors found per KLOC (that is, thousand lines of code) at different stages of the project might help to predict the correctness and reliability of the final system. Keystrokes required to carry out a particular transaction might help predict what the operator time to carry out the transaction is likely to be. Modularity, the degree to which the software is composed of self-contained manageable components, helps predict how easy changes to the final system will be.

1.13 Conclusion

This chapter has laid a foundation for the remainder of the book by defining what is meant by various terms such as 'software project' and 'management'. Among some of the more important points that have been made are the following.

- Projects are by definition non-routine and therefore more uncertain than normal undertakings.

- Software projects are similar to other projects, but have some attributes that present particular difficulties, for example, the relative invisibility of many of their products.
- A key factor in project success is having clear objectives. Different stakeholders in a project, however, are likely to have different objectives. This points to the need for a recognized overall project authority.
- For objectives to be effective, there must be practical ways of testing that the objectives have been met. Hence there is a need for measurement.
- Where projects involve many different people, effective channels of information have to be established. Having objective measures of success helps unambiguous communication among the various parties to a project.

1.14 Further exercises

1. List the problems you experienced when you carried out a recent IT-related assignment. Try to put these problems into some order of magnitude. For each problem, consider whether there was some way in which the problem could have been reduced by better organization and planning by yourself.
2. Identify the main types of personnel employed in an information systems department. For each stage of a typical IS development project, list the types of personnel who are likely to be involved.
3. A public library department is considering the implementation of a computer-based system to help administer book loans at libraries. Identify the stakeholders in such a project. What might be the objectives of such a project and how might the success of the project be measured in practical terms?
4. A manager is in charge of a sub-project of a larger project. The sub-project requires the transfer of paper documents into a computer-based document retrieval system and their subsequent indexing so that they can be accessed via key-words. Optical character readers are to be used for the initial transfer but the text then needs to be clerically checked and corrected by staff. The project is currently scheduled to take twelve months using permanent staff. A small budget is available to hire temporary staff in the case of staff absences through holidays, sickness or temporary transfer to other, more urgent, jobs. Discuss the control system that will be needed to control that sub-project.
5. In the above example, concerning document transfer and indexing, identify the strategic information that the manager might want to consider during the initial planning of the sub-project.
6. Suggest objectives for the following types of staff: (a) a data preparation clerk; (b) a programmer/analyst; (c) a computer software sales person; (d) a systems analyst; (e) a software project leader. In each case suggest two measures of efficiency or effectiveness.