7 Risk Management

Learning Objectives

- Identify the factors putting a project at risk
- Categorize and prioritize actions for risk elimination or containment
- Quantify the likely effects of risk on project timescales

7.1 Introduction

In Chapter 6 we saw how, at IOE, Amanda planned how the software for the new annual maintenance contracts application was to be produced. This included estimating how long each task would take – see Figure 6.7 and Table 6.2. Her plan was based on the assumption that three experienced programmers were available for the coding of modules A, B, C and D. However, suppose two developers then left for better-paid jobs, and so far only one replacement has been recruited, who happens to be a trainee.

In some work environments 'problems' in this context are referred to as 'issues'.

In the case of Brigette and the Brightmouth payroll implementation project, imagine that a payroll package has been purchased. However, a new requirement emerges that the payroll database should be accessed by a new application that calculates the staff costs for each course delivered by the college. Unfortunately, the purchased payroll application does not allow this access.

Amanda and Brigette will have to deal with these *problems* as part of the monitoring and control process that will be outlined in Chapter 9. In this chapter we consider whether the two project leaders could have foreseen that these problems were likely to occur and made plans to deal with them. In other words, could these problems have been identified as *risks*?

7.2 Risk

PM-BOK defines risk as '*an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives*'. PRINCE2, the UK government-sponsored project management standard, defines risk as '*the chance of exposure to the adverse consequences of future events*'. The two definitions differ, as the first includes situations where a future uncertainty actually works in our favour and presents us with an opportunity. We will return to this later in the chapter.

PM-BOK stands for Project Management Body of Knowledge, a project management standard published by the Project Management Institute in the USA.

The key elements of a risk follow.

- *It relates to the future* The future is inherently uncertain. Some things which seem obvious when a project is over, for example that the costs were underestimated or that a new technology was overly difficult to use, might not have been so obvious during planning.
- *It involves **cause** and **effect*** For example, a 'cost over-run' might be identified as a risk, but 'cost over-run' describes some damage, but does not say what causes it. Is it, for example, an inaccurate estimate of effort, the use of untrained staff, or a poor specification? Both the cause (or *hazard*), such as 'inexperienced staff', and a particular type of negative outcome, such as 'lower productivity', should be defined for each risk.

The ISPL risk model (formerly Euromethod) refers to hazards as 'situational factors'.

Exercise 7.1

Match the following causes – (a) to (d) – to their possible effects – (i) to (iv). The relationships are not necessarily one-to-one. Explain the reasons for each match.

Causes

(a) Staff inexperience

(b) Lack of top management commitment

(c) New technology

(d) Users uncertain of their requirements

Effects

(i) Testing takes longer than planned

(ii) Planned effort and time for activities exceeded

(iii) Project scope increases

(iv) Time delays in getting changes to plans agreed

The boundary between risk management and 'normal' software project management is hazy. For example, when we were selecting the best general approach to a project – see Chapter 4 – one consideration was the possible consequences of future adverse events. As will be seen in Chapter 13, most of the techniques used to assure the quality of software, such as reviews and testing, are designed to reduce the risk of faults in project deliverables. Risk management is not a self-contained topic within project management. The key role of risk management is considering uncertainty remaining after a plan has been formulated. Every plan is based on assumptions and risk management tries to plan for and control the situations where those assumptions become incorrect. Risk planning is carried out in Steps 3 and 6 (Figure 7.1).

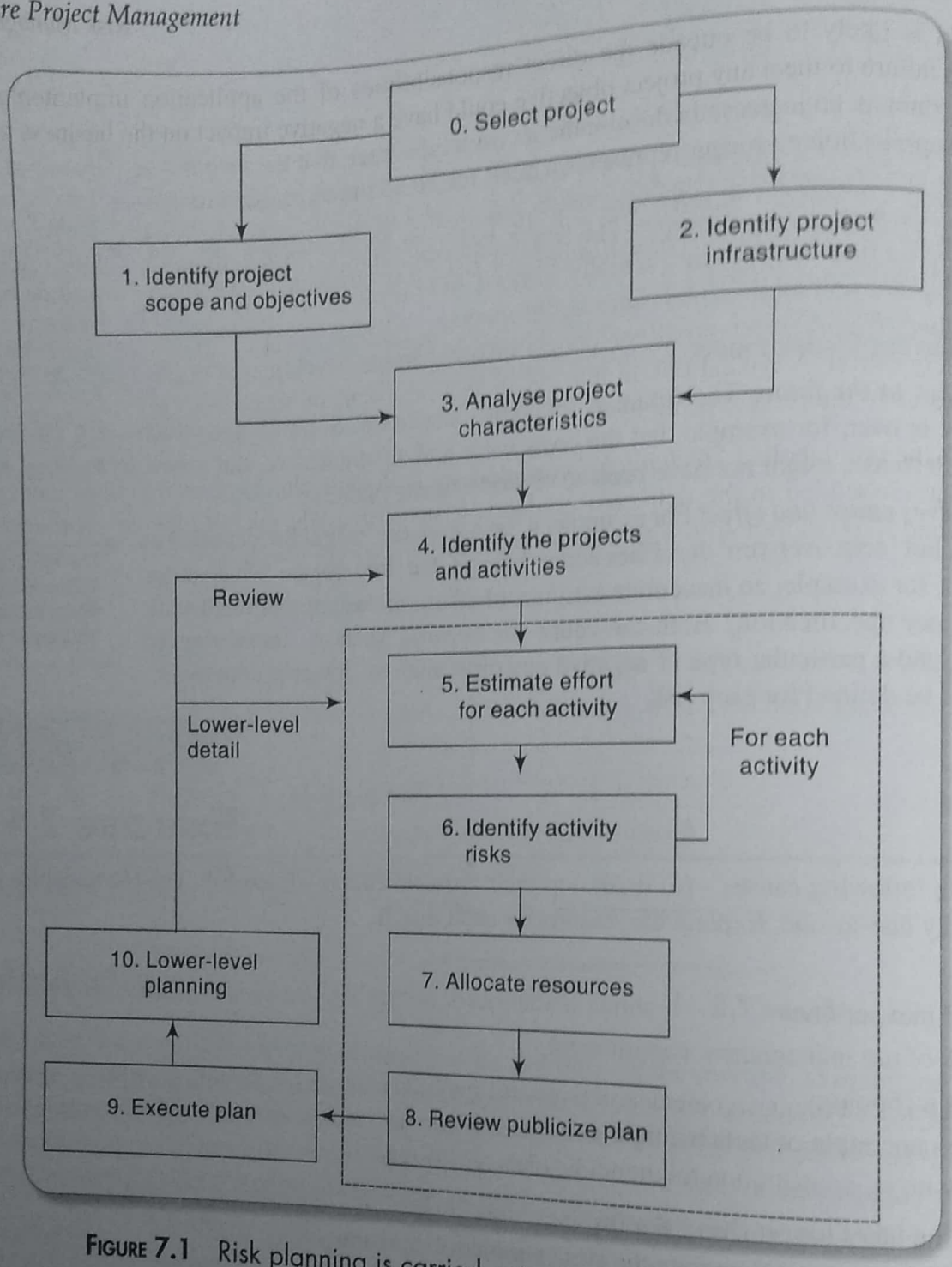


FIGURE 7.1 Risk planning is carried out primarily in Steps 3 and 6

7.3 Categories of Risk

An ICT project manager is normally given the objective of installing the required application by a specified deadline and within an agreed budget. Other objectives might be set, especially with regard to quality requirements. *Project risks* are those that could prevent the achievement of the objectives given to the project manager and the project team.

As we noted in Chapter 2, there could be risks that an application after successful implementation is a business failure. Thus if an e-commerce site is established to sell a product, the site might be correctly implemented, but customers fail to use the site because of the uncompetitive prices demanded. Dealing with these

business risks is likely to be outside the direct responsibilities of the application implementation team. However, the failure to meet any project objective could have a negative impact on the business case for the project. For example, an increase in development cost might mean that the income (or savings) generated by the delivered application no longer represents a good return on the increased investment.

Risks have been categorized in other ways. Kalle Lyytinen and his colleagues, for instance, have proposed a *sociotechnical model* of risk, a diagrammatic representation of which appears in Figure 7.2.

See K. Lyytinen, L. Mathiassen and J. Ropponen (1996) 'A framework for risk management' *Journal of Information Technology*, 11(4).

The box labelled '*Actors*' refers to all the people involved in the development of the application in question. A typical risk in this area is that high staff turnover leads to expertise of value to the project being lost.

In Figure 7.2, the box labelled '*Technology*' encompasses both the technology used to implement the application and that embedded in the delivered products. Risks here could relate to the appropriateness of the technologies and to possible faults within them, especially if they are novel.

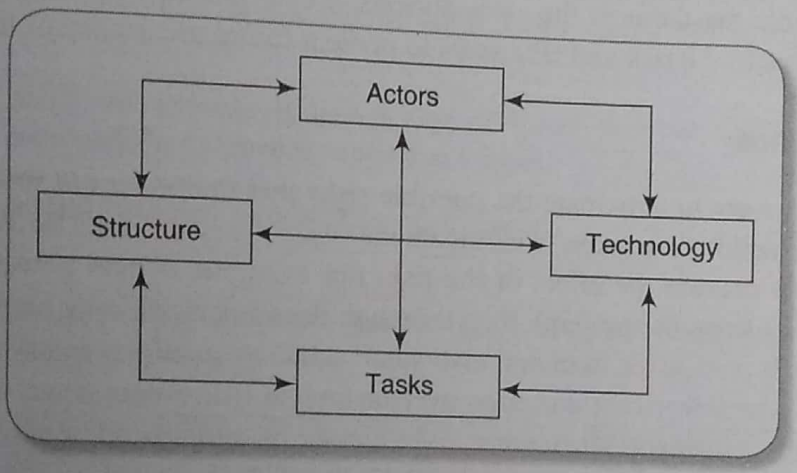


FIGURE 7.2 Lyytinen–Mathiassen–Ropponen risk framework

'*Structure*' describes the management structures and systems, including those affecting planning and control. For example, the implementation might need user participation in some tasks, but the responsibility for managing the users' contribution might not be clearly allocated.

'*Tasks*' relates to the work planned. For instance, the complexity of the work might lead to delays because of the additional time required integrate the large number of components.

In Figure 7.2 all boxes are interlinked. Risks often arise from the relationships between factors – for example between technology and people. If a development technology is novel then the developers might not be experienced in its use and delay results. The novelty of the new technology is really a characteristic of the developers: once they are used to the technology, it is no longer 'novel'.

Exercise 7.2

In the cases of the Brightmouth payroll implementation project and the IOE annual maintenance contracts development project, identify one risk for each of the four categories in Figure 7.2.

7.4 Risk Management Approaches

Risk management approaches can broadly be classified into reactive and proactive approaches. The latter approach is much more effective in risk handling and, therefore, used wherever possible. In the following, we briefly discuss these two approaches.

Reactive approaches

Reactive approaches take no action until an unfavourable event occurs. Once an unfavourable event occurs, these approaches try to contain the adverse effects associated with the risk and take steps to prevent future occurrence of the same risk events. An example of such a risk management strategy can be the following. Consider a project in which the server hosting the project data crashes. Once this risk event has occurred, the team members may put best effort to recover the data and also initiate the practice of taking regular backups, so that in future such a risk event does not recur. It is similar to calling the emergency fire-fighting service once a fire has been noticed, and then installing fire-fighting equipment in all the rooms of the building to be able to instantly handle fire the next time it is noticed. It can be seen that the main objective of this is to minimize the damage due to the risk and take steps to prevent future recurrence of the risk.

Proactive approaches

The proactive approaches try to anticipate the possible risks that the project is susceptible to. After identifying the possible risks, actions are taken to eliminate the risks. If a risk cannot be avoided, these approaches suggest making plans to contain the effect of the risk. For example, if man power turnover is anticipated (i.e. some personnel may leave the project), then thorough documentation may be planned. Also, more than one developer may work on a work item and also some stand-by man power may be planned. Obviously, proactive approaches incur lower cost and time overruns when risk events occur and, therefore, are much more preferred by teams. However, when some risks cannot be anticipated, a reactive approach is usually followed.

7.5 A Framework for Dealing with Risk

Planning for risk includes these steps:

(i) Risk identification

(ii) Risk analysis and prioritization

(iii) Risk planning

(iv) Risk monitoring

Steps (i) to (iii) above will probably be repeated. When risks that could prevent a project success are identified, plans can be made to reduce or remove their threat. The plans are then reassessed to ensure that the original risks are reduced sufficiently and no new risks inadvertently introduced. Take the risk that staff inexperience with a new technology could lead to delays in software development. To reduce this risk, consultants expert in the new technology might be recruited. However, the use of consultants might introduce the new risk that knowledge about the new technology is not transferred to the permanent staff, making subsequent software maintenance problematic. Having identified this new risk, further risk reduction activities can be planned.

7.6 Risk Identification

The two main approaches to the identification of risks are the use of *checklists* and *brainstorming*.

Checklists are simply lists of the risks that have been found to occur regularly in software development projects. A specialized list of software development risks by Barry Boehm appears in Table 7.1 in a modified version. Ideally a group of representative project stakeholders examines a checklist identifying risks applicable to their project. Often the checklist suggests potential countermeasures for each risk.

Table 7.1 Software project risks and strategies for risk reduction

Risk	Risk reduction techniques
Personnel shortfalls	Staffing with top talent; job matching; teambuilding; training and career development; early scheduling of key personnel
Unrealistic time and cost estimates	Multiple estimation techniques; design to cost; incremental development; recording and analysis of past projects; standardization of methods
Developing the wrong software functions	Improved software evaluation; formal specification methods; user surveys; prototyping; early user manuals
Developing the wrong user interface	Prototyping; task analysis; user involvement
Gold plating	Requirements scrubbing; prototyping; cost–benefit analysis; design to cost
Late changes to requirements	Stringent change control procedures; high change threshold; incremental development (deferring changes)
Shortfalls in externally supplied components	Benchmarking; inspections; formal specifications; contractual agreements; quality assurance procedures and certification
Shortfalls in externally performed tasks	Quality assurance procedures; competitive design or prototyping; contract incentives
Real-time performance shortfalls	Simulation; benchmarking; prototyping; tuning; technical analysis
Development technically too difficult	Technical analysis; cost–benefit analysis; prototyping; staff training and development

This top ten list of software risks is based on one presented by Barry Boehm in his *Tutorial on Software Risk Management*, IEE Computer Society, 1989.

Project management methodologies, such PRINCE2, often recommend that on completion of a project a review identifies any problems during the project and the steps that were (or should have been) taken to resolve or avoid them. These problems could in some cases be added to an organizational risk checklist for use with new projects.

The 'lessons learnt' report differs from a 'post implementation review' (PIR). It is written on project completion and focuses on project issues. A PIR, produced when the application has been operational for some time, focuses on business benefits.

Brainstorming

Ideally, representatives of the main stakeholders should be brought together once some kind of preliminary plan has been drafted. They then identify, using their individual knowledge of different parts of the project, the problems that might occur. This collaborative approach may generate a sense of ownership in the project.

'Brainstorming' is also mentioned in Chapter 13 in connection with quality circles.

Brainstorming might be used with Brigette's Brightmouth payroll implementation project as she realizes that there are aspects of college administration of which she is unaware. She therefore suggests to the main stakeholders in the project, who include staff from the finance office and the personnel office, that they meet and discuss where the risks facing the project lie.

7.7 Risk Assessment

A common problem with risk identification is that a list of risks is potentially endless. A way is needed of distinguishing the damaging and likely risks. This can be done by estimating the *risk exposure* for each risk using the formula:

$$\textit{risk exposure} = (\textit{potential damage}) \times (\textit{probability of occurrence})$$

Using the most rigorous – but not necessarily the most practical – approach, the potential damage would be assessed as a money value. Say a project depended on a data centre vulnerable to fire. It might be estimated that if a fire occurred a new computer configuration could be established for £500,000. It might also be estimated that where the computer is located there is a 1 in 1000 chance of a fire actually happening, that is a probability of 0.001.

The risk exposure in this case would be:

$$£500{,}000 \times 0.001 = £500$$

A crude way of understanding this value is as the minimum sum an insurance company would require as a premium. If 1000 companies, all in the same position, each contributed £500 to a fund then, when the 1 in 1000 chance of the fire actually occurred, there would be enough money to cover the cost of recovery.

Exercise 7.3

What conditions would have to exist for the risk pooling arrangement described above to work?

The calculation of risk exposure above assumes that the amount of damage sustained will always be the same. However, it is usually the case that there could be varying amounts of damage. For example, as software development proceeds, more software is created, and more time would be needed to re-create it if it were lost.

With some risks, there could be not only damage but also gains. The testing of a software component is scheduled to take six days, but is actually done in three days. A team leader might therefore feel justified in producing a probability chart like the one in Figure 7.3. This shows the probability of a task being completed in four days (5%), then five days (10%), and so on. The accumulated probability for the seventh day (65%) means that there is a 65% chance that the task will be finished on or before the seventh day.

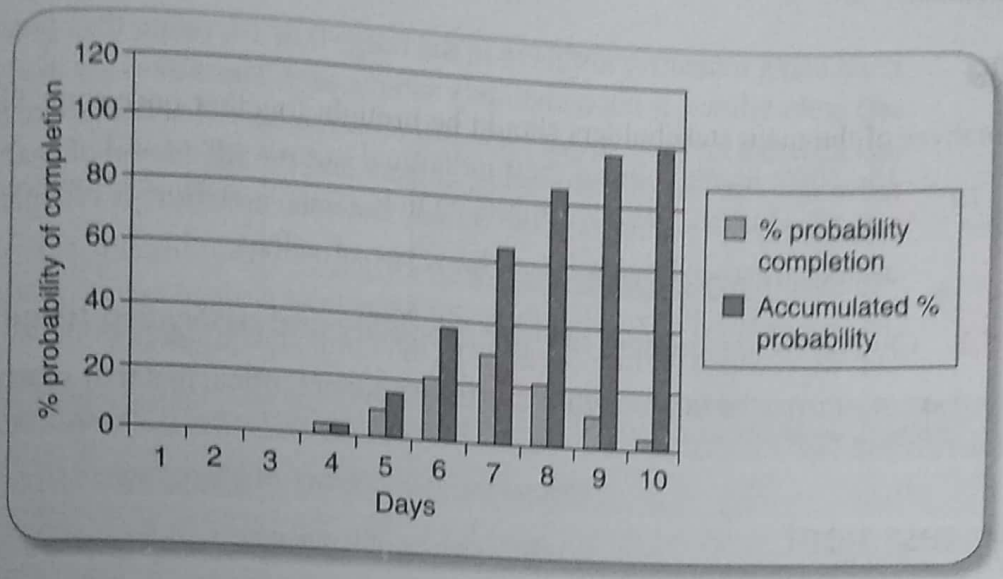


FIGURE 7.3 Probability chart

Clients would almost certainly insist we pick one of the days as the target. This target could be 'aggressive', for instance only five days in the above scenario, but with an 85% chance of failure according to the chart. A safer estimate would be eight days which would have a probability of failure of only 15%. We will return to this point later on in this chapter.

In Figure 7.3 the 'loss' is effectively being measured in days rather than money. In this context, days, or some other unit of personal effort, is often used as a *surrogate* for a financial loss.

Most managers resist very precise estimates of loss or of the probability of something occurring, as such figures are usually guesses. Barry Boehm has suggested that, because of this, both the risk losses and the probabilities be assessed using relative scales in the range 0 to 10. The two figures could then be multiplied together to get a notional risk exposure. Table 7.2 provides an example, based on Amanda's IOE group accounts project, of where this has been done. This value could be used to prioritize the importance of risks, although more sophisticated risk calculations are not possible.

TABLE 7.2 Part of Amanda's risk exposure assessment

Ref	Hazard	Likelihood	Impact	Risk
R1	Changes to requirements specification during coding	8	8	64
R2	Specification takes longer than expected	3	7	21
R3	Significant staff sickness affecting critical path activities	5	7	35
R4	Significant staff sickness affecting non-critical activities	10	3	30
R5	Module coding takes longer than expected	4	5	20
R6	Module testing demonstrates errors or deficiencies in design	4	8	32

Boehm suggests that planners focus attention on the 10 risks with the highest risk exposure scores. For smaller projects – including the final-year projects of computing students – the focus could be on a smaller number of risks.

Even using indicative numbers in the range 0 to 10, rather than precise money values and probabilities, is not completely satisfactory. The values are likely to be subjective, and different analysts might pick different numbers. Another approach is to use qualitative descriptions of the possible impact and the likelihood of each risk – see Tables 7.3 and 7.4 for examples. Consistency between assessors is facilitated by associating each qualitative description with a range of values.

See P. Goodwin and G. Wright (2004) *Decision Analysis for Management Judgement*, Wiley, for further discussion of this issue.

Table 7.3 Qualitative descriptors of risk probability and associated range values

Probability level	Range
High	Greater than 50% chance of happening
Significant	30–50% chance of happening
Moderate	10–29% chance of happening
Low	Less than 10% chance of happening

Table 7.4 Qualitative descriptors of impact on cost and associated range values

Impact level	Range
High	More than 30% above budgeted expenditure
Significant	20 to 29% above budgeted expenditure
Moderate	10 to 19% above budgeted expenditure
Low	Within 10% of budgeted expenditure.

In Table 7.4, the potential amount of damage has been categorized in terms of its impact on *project costs*. Other tables could show the impact of risks on *project duration* or on the *quality of the project deliverables*.

To some extent, the project manager, in conjunction with the project sponsor, can choose whether the damage inflicted by a risk affects cost, duration or the quality of deliverables. In Amanda's list of risks in Table 7.2, R5 refers to the coding of modules taking longer than planned. This would have an impact on both the duration of the project and the costs, as more staff time would be needed. A response might be adding software developers and splitting the remaining development work between them. This will increase costs, but could save the planned completion date. Another option is to save both duration and staff costs by reducing software testing before the software is released. This is likely to be at the price of decreased quality in the project deliverable.

Where the potential damage and likelihood of a risk are defined by qualitative descriptors, the risk exposure cannot be calculated by multiplying the two factors together. In this case, the risk exposure is indicated by the position of the risk in a matrix – see Figure 7.4. These matrices have variously been called *probability impact grids* or *summary risk profiles*.

In Figure 7.4, some of the cells in the top right of the matrix have been zoned off by a *tolerance line*. Risks that appear within this zone have a degree of seriousness that calls for particular attention.

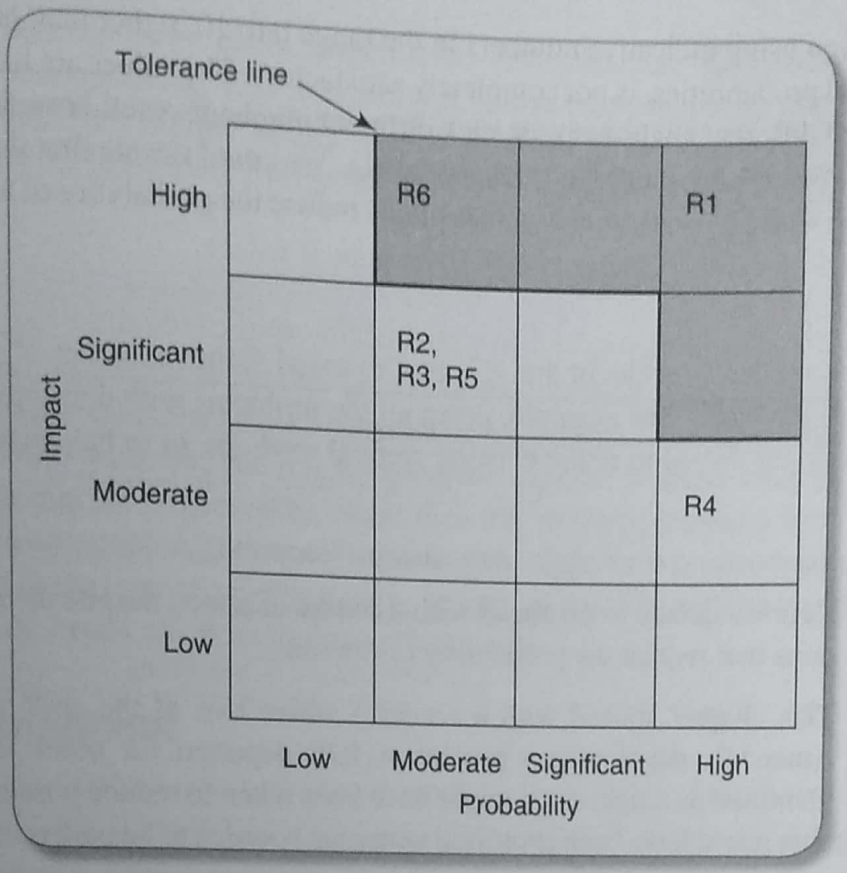


FIGURE 7.4 Probability impact matrix

Chapter 5 stressed the need for frequent reassessment of effort and duration estimates during a project. This applies to risk exposure as well, as some risks apply only at certain stages. A risk might be that key users are unavailable when needed to supply details of their requirements. As requirements are gathered, so this risk will diminish until it is no longer significant. In general, the element of uncertainty will lessen as a project progresses and more is learnt by the developers about user requirements and any new technology. This would be reflected in lower risk probabilities. On the other hand, the potential damage will tend to increase as the amount invested in the project grows. If you type a substantial report using a word processor and neglect to take back-ups, as each day adds more text to the report, it also adds to the number of days needed to re-key the report in the event of file loss.

The term *risk proximity* is used to describe this attribute of risk.

7.8 Risk Planning

Having identified the major risks and allocated priorities, the task is to decide how to deal with them. The choices discussed will be:

- Risk acceptance
- Risk avoidance
- Risk reduction and mitigation
- Risk transfer

Risk acceptance

This is the do-nothing option. We will already, in the risk prioritization process, have decided to ignore some risks in order to concentrate on the more likely or damaging. We could decide that the damage inflicted by some risks would be less than the costs of action that might reduce the probability of a risk happening.

Risk avoidance

Some activities may be so prone to accident that it is best to avoid them altogether. If you are worried about sharks then don't go into the water. For example, given all the problems with developing software solutions from scratch, managers might decide to retain existing clerical methods, or to buy an off-the-shelf solution.

Risk reduction

Here we decide to go ahead with a course of action despite the risks, but take precautions that reduce the probability of the risk.

It must be appreciated that each risk reduction action is likely to involve some cost. This is discussed in the next section.

This chapter started with a scenario where two of the staff scheduled to work on Amanda's development project at IOE departed for other jobs. If this has been identified as a risk, steps might have been taken to reduce possible departures of staff. For instance, the developers might have been promised generous bonuses to be paid on successful completion of the project.

Recall that Brigette had a problem at Brightmouth College: after the purchase of the payroll package, a requirement for the payroll database to be accessed by another application was identified. Unfortunately, the application that had been bought did not allow such access. An alternative scenario might have been that Brigette identified this as a possible risk early on in the project. She might have come across Richard Fairley's four COTS (commercial off-the-shelf) software acquisition risks – see Table 7.5 – where one risk is difficulty in integrating the data formats and communication protocols of different applications. Brigette might have specified that the selected package must use a widely accepted data management system like Oracle that allows easier integration.

Table 7.5 Fairley's four commercial off-the-shelf (COTS) software acquisition risks

Integration	Difficulties in integrating the data formats and communication protocols of different applications.
Upgrading	When the supplier upgrades the package, the package might no longer meet the users' precise requirements. Sticking with the old version could mean losing the supplier's support for the package.
No source code	If you want to enhance the system, you might not be able to do so as you do not have access to the source code.
Supplier failures or buyouts	The supplier of the application might go out of business or be bought out by a rival supplier.

See R. Fairley (1994) 'Risk management for software projects' *IEEE Software* 11(3) 57–67.

Risk mitigation can sometimes be distinguished from risk reduction. *Risk reduction* attempts to reduce the likelihood of the risk occurring. *Risk mitigation* is action taken to ensure that the impact of the risk is lessened when it occurs. For example, taking regular back-ups of data storage would reduce the impact of data corruption but not its likelihood. Mitigation is closely associated with contingency planning which is discussed presently.

Risk transfer

In this case the risk is transferred to another person or organization. With software projects, an example of this would be where a software development task is outsourced to an outside agency for a fixed fee. You might expect the supplier to quote a higher figure to cover the risk that the project takes longer than the 'average' expected time. On the other hand, a well-established external organization might have productivity advantages as its developers are experienced in the type of development to be carried out. The need to compete with other software development specialists would also tend to drive prices down.

Risk transfer is what effectively happens when you buy insurance.

7.9 Risk Management

Contingency

Risk reduction activities would appear to have only a small impact on reducing the probability of some risks, for example staff absence through illness. While some employers encourage their employees to adopt a healthy lifestyle, it remains likely that some project team members will at some point be brought down by minor illnesses such as flu. These kinds of risk need a *contingency plan*. This is a planned action to be carried out if the particular risk materializes. If a team member involved in urgent work were ill then the project manager might draft in another member of staff to cover that work.

The preventative measures that were discussed under the 'Risk reduction' heading above will usually incur some cost regardless of the risk materializing or not. The cost of a contingency measure will only be incurred if the risk actually materializes. However, there may be some things that have to be done in order for the contingency action to be feasible. An obvious example is that back-ups of a database have to be taken if the contingency action when the database is corrupted is to restore it from back-ups. There would be a cost associated with taking the back-ups.

Exercise 7.4

In the case above where staff could be absent through illness, what preconditions could facilitate contingency actions such as getting other team members to cover on urgent tasks? What factors would you consider in deciding whether these preparatory measures would be worthwhile?

Deciding on the risk actions

Five generic responses to a risk have been discussed above. For each actual risk, however, specific actions have to be planned. In many cases experts have produced lists recommending practical steps to cope with the likelihood of particular risks; see, for example, Boehm's 'top ten' software engineering risks in Table 7.1.

Whatever the countermeasures that are considered, they must be cost-effective. On those occasions where a risk exposure value can be calculated as a financial value using the (*value of damage*) × (*probability of occurrence*) formula – recall Section 7.7 – the cost-effectiveness of a risk reduction action can be assessed by calculating the *risk reduction leverage* (*RRL*).

$$risk\ reduction\ leverage = (RE_{before} - RE_{after})/(cost\ of\ risk\ reduction)$$

RE_{before} is the risk exposure, as explained in Section 7.7, before risk reduction actions have been taken. RE_{after} is the risk exposure after taking the risk reduction action. An RRL above 1.00 indicates that the reduction in risk exposure achieved by a measure is greater than its cost. To take a rather unrealistic example, it might cost £200,000 to replace a hardware configuration used to develop a software application. There is a 1% chance of a fire (because of the particular location of the installation, say). The risk exposure would be 1% of £200,000, that is £2,000. Installing fire alarms at a cost of £500 would reduce the chance of fire to 0.5%. The new risk exposure would be £1,000, a reduction of £1,000 on the previous exposure. The RRL would be (2000 – 1000)/500, that is 2.0, and the action would therefore be deemed worthwhile.

Earlier in this chapter, we likened risk exposure to the amount you might pay to an insurance company to cover a risk. To continue the analogy, an insurance company in the above example might be willing to reduce the premium you pay to have cover against fire from £2,000 to £1,000 if you installed fire alarms. As the fire alarms would cost you only £500 and save £1,000, the cost would clearly be worthwhile.

Exercise 7.5

Assume that the likelihood of one of your valuable team members leaving the project midway is 0.5. In case the member actually leaves, there is a 25% chance that the project would miss the delivery date. You consider the customer's consequent displeasure to be equivalent to £50,000 in monetary terms. To counter the risk, you can recruit a fresh engineer at a salary of £2000 per month for six months, to essentially act as a back-up for the valuable team member. Also, assume that the contribution of the back-up engineer to the project, if the regular engineer does not leave, would be 0.2 of the employment duration. After employing the back-up engineer, the probability of missing the project deadline is expected to be only about 10%. Would it be a good idea to employ the back-up engineer?

Creating and maintaining the risk register

When the project planners have picked out and examined what appear to be the most threatening risks to the project, they need to record their findings in a *risk register*. The typical content of such a register is shown in Figure 7.5. After work starts on the project more risks will emerge and be added to the register. At regular intervals, probably as part of the project control life cycle described in Chapter 9, the risk register should be reviewed and amended. Many risks threaten just one or two activities, and when the project staff have completed these the risk can then be 'closed' as no longer relevant. In any case, as noted earlier, the probability and impact of a risk are likely to change during the course of the project.

7.10 Evaluating Risks to the Schedule

In Section 7.7, we showed a probability chart – Figure 7.3. This illustrated the point that a forecast of the time needed to do a job is most realistically presented as a graph of likelihood of a range of figures, with the most

RISK RECORD

Risk id		Risk title		
Owner		Date raised	Status	

Risk description

Impact description

Recommended risk mitigation

Probability/impact values

	Probability	Impact		
		Cost	Duration	Quality
Pre-mitigation				
Post-mitigation				

Incident/action history

Date	Incident/action	Actor	Outcome/comment

FIGURE 7.5 Risk register page

likely duration as the peak and the chances of the job taking longer or shorter shown as curves sloping down on either side of the peak. Thus we can show that a job might take five days but that there is a small chance it might need four or six days, and a smaller chance of three or seven days, and so on. If a task in a project takes longer than planned, we might hope that some other task might take less and thus compensate for this delay. In the following sections we will examine PERT, a technique which takes account of the uncertainties in the durations of activities within a project. We will also touch upon Monte Carlo simulation, which is a more powerful and flexible tool that tackles the same problem.

A drawback to the application of methods like PERT is that in practice there is a tendency for developers to work to the schedule even if a task could be completed more quickly. Even if tasks are completed earlier than planned, project managers are not always quick to exploit the opportunities to start subsequent activities earlier than scheduled. Critical chain management will be explored as a way of tackling this problem.

7.11 Boehm's Top 10 Risks and Counter Measures

Boehm has identified the top 10 risks that a typical project suffers from and has recommended a set of countermeasures for each. We briefly review these in the following.

Barry W. Boehm, 'Software Risk Management:Principles and Practices,' *IEEE Software*, Volume 8, Issue 1, January 1991.

1. **Personnel shortfall:** This risk concerns shortfall of project personnel. The shortfall may show up as either project personnel may lack some specific competence required for the project tasks or personnel leaving the project (called manpower turnover) before project completion. The countermeasures suggested include staffing with top talent, job matching, team building, and cross-training of personnel.
2. **Unrealistic schedules and budgets:** The suggested counter measures include the project manager working out the detailed milestones and making cost and schedule estimations based on it. Other counter measures are incremental development, software reuse, and requirements scrubbing. It may be mentioned that requirements scrubbing involves removing the overly complex and unimportant requirements, in consultation with the customers.
3. **Developing the wrong functions:** The suggested countermeasures include user surveys and user participation, developing prototypes and eliciting user feedback, and early production users' manuals and getting user feedback on it.
4. **Developing the wrong user interface:** The countermeasures suggested for this risk include prototyping, scenarios and task analysis, and user participation.
5. **Gold-plating:** Gold-plating as discussed in Chapter 1, concerns development of features that the team members consider nice to have and, therefore, decide to develop those even though the customer has not expressed any necessity for those. The countermeasures suggested for this risk includes requirements scrubbing, prototyping and cost-benefit analysis.
6. **Continuing stream of requirements changes:** The countermeasures suggested for this risk include incremental development, high change threshold and information hiding.
7. **Shortfalls in externally-furnished components:** This concerns the risk that the components developed by third party are not up to the mark. The countermeasures suggested for this risk include benchmarking, inspections, reference checking and compatibility analysis.
8. **Shortfalls in externally performed tasks**: This concerns the risk that the work performed by the contractors may not be up to the mark. The countermeasures suggested for this risk include reference checking, pre-award audits, award-fee contracts, competitive design or prototyping and team building.
9. **Real-time performance shortfalls:** The countermeasures suggested for this risk include simulation, benchmarking, modelling, prototyping, instrumentation and tuning.
10. **Straining computer science capabilities**: The countermeasures suggested for this risk include technical analysis, cost-benefit analysis and prototyping.

Risk Mitigation, Monitoring, and Management (RMMM) Plan

It is usually advisable for the project manager to develop a risk mitigation, monitoring and management (RMMM) plan for a project. An important component of this document is a *risk table*. Each row of the table contains the name of the risk, its probability and its impact on the project. For each risk in the risk table, the specific conditions or events that need to be monitored to check whether the risk has actually occurred is mentioned. The possible ways in which the risk can be avoided (mitigation) is also documented. A contingency plan to contain the effect of the risk is also documented.

7.12 Applying the PERT Technique

Using PERT to evaluate the effects of uncertainty

PERT was developed to take account of the uncertainty surrounding estimates of task durations. It was developed in an environment of expensive, high-risk and state-of-the-art projects – not that dissimilar to many of today's large software projects.

The method is very similar to the CPM technique (indeed many practitioners use the terms PERT and CPM interchangeably) but, instead of using a single estimate for the duration of each task, PERT requires three estimates.

- *Most likely time*: the time we would expect the task to take under normal circumstances. We shall identify this by the letter m.
- *Optimistic time*: the shortest time in which we could expect to complete the activity, barring outright miracles. We shall use the letter a for this.
- *Pessimistic time*: the worst possible time, allowing for all reasonable eventualities but excluding 'acts of God and warfare' (as they say in most insurance exclusion clauses). We shall call this b.

PERT (Program Evaluation and Review Technique) was published in the same year as CPM. Developed for the Fleet Ballistic Missiles Program, it is said to have saved considerable time in development of the Polaris missile.

PERT then combines these three estimates to form a single expected duration, t_e, using the formula

$$t_e = \frac{a + 4m + b}{6}$$

Exercise 7.6

Table 7.6 provides additional activity duration estimates for the network shown in Figure 6.29. There are new estimates for a and b and the original activity duration estimates have been used as the most likely times, m. Calculate the expected duration, t_e, for each activity.

TABLE 7.6 PERT activity time estimates

Activity	Optimistic (a)	Activity durations (weeks). Most likely (m)	Pessimistic (b)
A	5	6	8
B	3	4	5

C	2	3	3
D	3.5	4	5
E	1	3	4
F	8	10	15
G	2	3	4
H	2	2	2.5

Using expected durations

The expected durations are used to carry out a forward pass through a network, using the same method as the CPM technique. In this case, however, the calculated event dates are not the earliest possible dates but the dates by which we expect to achieve those events.

Exercise 7.7

Before reading further, use your calculated expected activity durations to carry out a forward pass through the network (Figure 6.29) and verify that the project duration is 13.5 weeks. What does an expected duration of 13.5 weeks mean in terms of the completion date for the project?

The PERT network illustrated in Figure 7.6 indicates that we expect the project to take 13.5 weeks. In Figure 7.6 we have used an activity-on-arrow network as this form of presentation makes it easier to separate visually the estimated activity data (expected durations and, later, their standard deviations) from the calculated data (expected completion dates and target completion dates). The method can, of course, be equally well supported by activity-on-node diagrams.

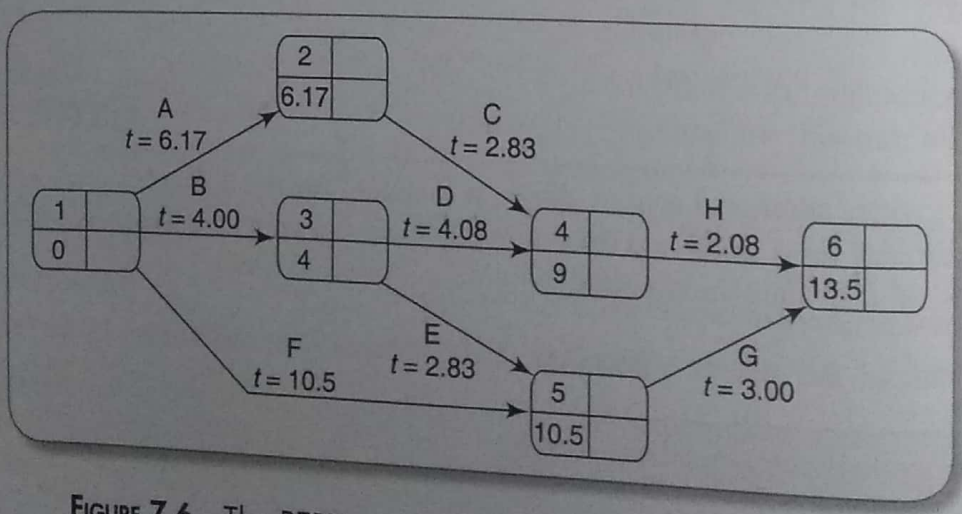
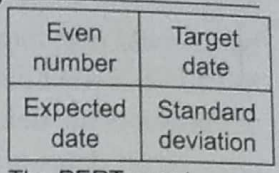


FIGURE 7.6 The PERT network after the forward pass

Unlike the CPM approach, the PERT method does not indicate the earliest date by which we could complete the project but the expected (or most likely) date. An advantage of this approach is that it places an emphasis on the uncertainty of the real world. Rather than being tempted to say '*the completion date for the project is. . .*' we are led to say '*we expect to complete the project by. . .*'.



The PERT event labelling convention adopted here indicates event number and its target date along with the calculated values for expected time and standard deviation.

It also focuses attention on the uncertainty of the estimation of activity durations. Requesting three estimates for each activity emphasizes the fact that we are not certain what will happen – we are forced to take into account the fact that estimates are approximate.

Activity standard deviations

A quantitative measure of the degree of uncertainty of an activity duration estimate may be obtained by calculating the standard deviation s of an activity time, using the formula

$$s = \frac{b - a}{6}$$

This standard deviation formula is based on the rationale that there are approximately six standard deviations between the extreme tails of many statistical distributions.

The activity standard deviation is proportional to the difference between the optimistic and pessimistic estimates, and can be used as a ranking measure of the degree of uncertainty or risk for each activity. The activity expected durations and standard deviations for our sample project are shown in Table 7.7.

The likelihood of meeting targets

The main advantage of the PERT technique is that it provides a method for estimating the probability of meeting or missing target dates. There might be only a single target date – the project completion – but we might wish to set additional intermediate targets.

TABLE 7.7 Expected times and standard deviations

Activity	Activity durations (weeks)				
	Optimistic (a)	Most likely (m)	Pessimistic (b)	Expected (t_e)	Standard deviation (s)
A	5	6	8	6.17	0.50
B	3	4	5	4.00	0.33
C	2	3	3	2.83	0.17
D	3.5	4	5	4.08	0.25
E	1	3	4	2.83	0.50
F	8	10	15	10.50	1.17
G	2	3	4	3.00	0.33
H	2	2	2.5	2.08	0.08

Suppose that we must complete the project within 15 weeks at the outside. We expect it will take 13.5 weeks but it could take more or, perhaps, less. In addition, suppose that activity C must be completed by week 10, as it is to be carried out by a member of staff who is scheduled to be working on another project, and that event 5 represents the delivery of intermediate products to the customer, which must take place by week 10. These three target dates are shown on the PERT network in Figure 7.7.

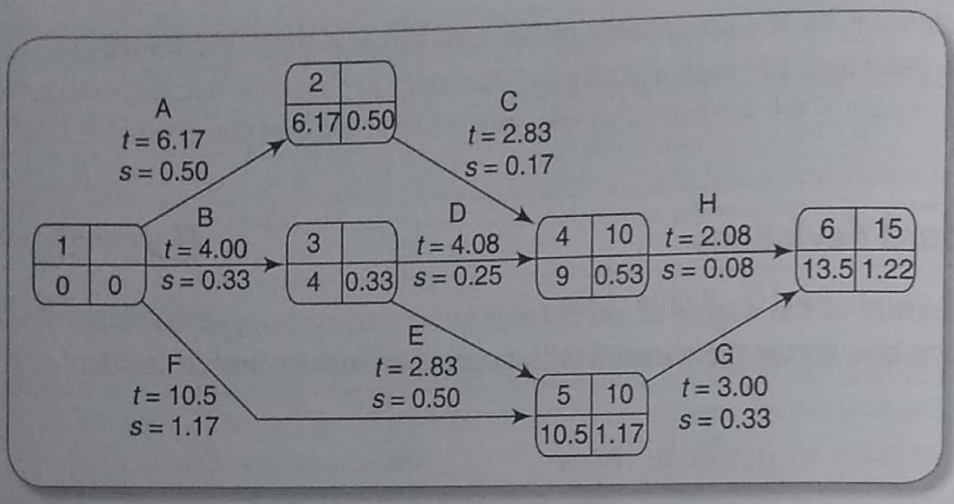


FIGURE 7.7 The PERT network with three target dates and calculated event standard deviations

The PERT technique uses the following three-step method for calculating the probability of meeting or missing a target date:

- calculate the standard deviation of each project event;
- calculate the z value for each event that has a target date;
- convert z values to a probabilities.

Calculating the standard deviation of each project event

The square of the standard deviation is known as the variance. Standard deviations may not be added together but variances may.

Standard deviations for the project events can be calculated by carrying out a forward pass using the activity standard deviations in a manner similar to that used with expected durations. There is, however, one small difference – to add two standard deviations we must add their squares and then find the square root of the sum. Exercise 7.8 illustrates the technique. One practical outcome of this is that the contingency time to be allocated to a sequence of activities as a whole would be less than the sum of the contingency allowances for each of the component activities. This has implications that can be exploited in critical chain project management, which are discussed in the next section.

Exercise 7.8

The standard deviation for event 3 depends solely on that of activity B. The standard deviation for event 3 is therefore 0.33.

For event 5 there are two possible paths, B + E or F. The total standard deviation for path B + E is $\sqrt{(0.33^2 + 0.50^2)} = 0.6$ and that for path F is 1.17; the standard deviation for event 5 is therefore the greater of the two, 1.17.

Verify that the standard deviations for each of the other events in the project are as shown in Figure 7.7.

Calculating the z values

The z value is calculated for each node that has a target date. It is equivalent to the number of standard deviations between the node's expected and target dates. It is calculated using the formula

$$z = \frac{T - t_e}{s}$$

where t_e is the expected date and T the target date.

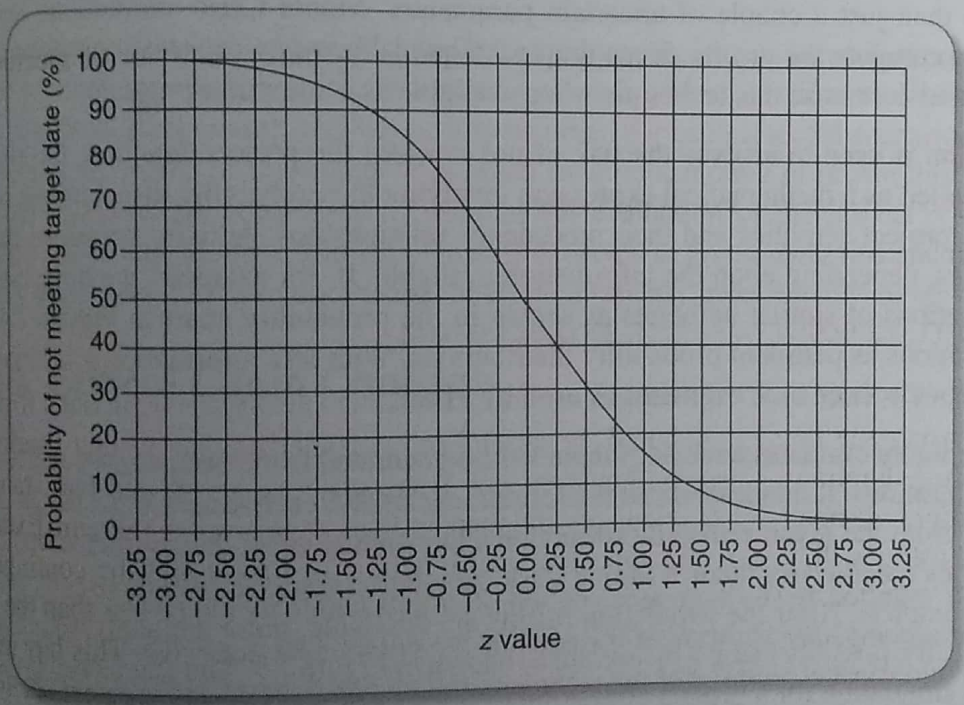
Exercise 7.9

The z value for event 4 is $(10 - 9.00)/0.53 = 1.8867$.

Calculate the z values for the other events with target dates in the network shown in Figure 7.7.

Converting z values to probabilities

A z value may be converted to the probability of not meeting the target date by using the graph in Figure 7.8.



This graph is the equivalent of tables of z values, also known as standard normal deviates, which may be found in most statistics textbooks.

FIGURE 7.8 The probability of obtaining a value within z standard deviations of the mean for a normal distribution

Exercise 7.10

The z value for the project completion (event 6) is 1.23. Using Figure 7.8 we can see that this equates to a probability of approximately 11%, that is, there is an 11% risk of not meeting the target date of the end of week 15.

Find the probabilities of not achieving events 4 or 5 by their target dates of the end of week 10.
What is the likelihood of completing the project by week 14?

Advantages of PERT

We have seen that by requesting multi-valued activity duration estimates and calculating expected dates, PERT focuses attention on the uncertainty of forecasting. We can use the technique to calculate the standard deviation for each task and use this to rank them according to their degree of risk. Using this ranking, we can see, for example, that activity F is the one regarding which we have greatest uncertainty, whereas activity C should, in principle, give us relatively little cause for concern.

If we use the expected times and standard deviations for forward passes through the network we can, for any event or activity completion, estimate the probability of meeting any set target. In particular, by setting target dates along the critical path, we can focus on those activities posing the greatest risk to the project's schedule.

7.13 Monte Carlo Simulation

As an alternative to the PERT technique, we can use Monte Carlo simulation approach. Monte Carlo simulation are a class of general analysis techniques that are valuable to solve any problem that is complex, nonlinear, or involves more than just a couple of uncertain parameters. Monte Carlo simulations involve repeated random sampling to compute the results. Since this technique is based on repeated computation of random numbers, it becomes easier to use this technique when available as a computer program.

When Monte Carlo simulation is used to analyse the risk of not meeting the project deadline, the project completion time is first modelled as a mathematical expression involving the probability distributions of the completion times of various project activities and their precedence relationships. Activity durations can be specified in a variety of forms, depending upon the information available. If, for example, we have historic data available about the durations of similar activities as shown in the probability chart in Figure 7.4, we might be able to specify durations as pertinent probability distributions. With less information available, we should, at least, be able to provide three time estimates as used by PERT.

Monte Carlo simulation essentially evaluates a range of input values generated from the specified probability distributions of the activity durations. It then calculates the results repeatedly; each time using a different set of random values generated from the given probability functions. Depending upon the number of probabilistic parameters and the ranges specified for them, a Monte Carlo simulation could involve thousands or even millions of calculations to complete. After the simulation results are available, these are analysed, summarized and represented graphically, possibly using a histogram as shown in Figure 7.9. The main steps involved in carrying out Monte Carlo simulation for a project consisting of n activities are as follows:

- *Step 1:* Express the project completion time in terms of the duration of the n activities ($x_{i,\ i=1,\ n}$ and their dependences as a precedence graph, $d = f(x_1, x_2, \ldots, x_n)$.
- *Step 2:* Generate a set of random inputs, $x_{i1}, x_{i2}, \ldots, x_{in}$ using specified probability distributions.
- Step 3: Evaluate the project completion time expression and store the result in d_i.
- *Step 4:* Repeat Steps 2 and 3 for the specified number of times.
- *Step 5:* Analyze the results $d_{i,\ i=1,n}$; summarize and display using a histogram as the one shown in Figure 7.9.

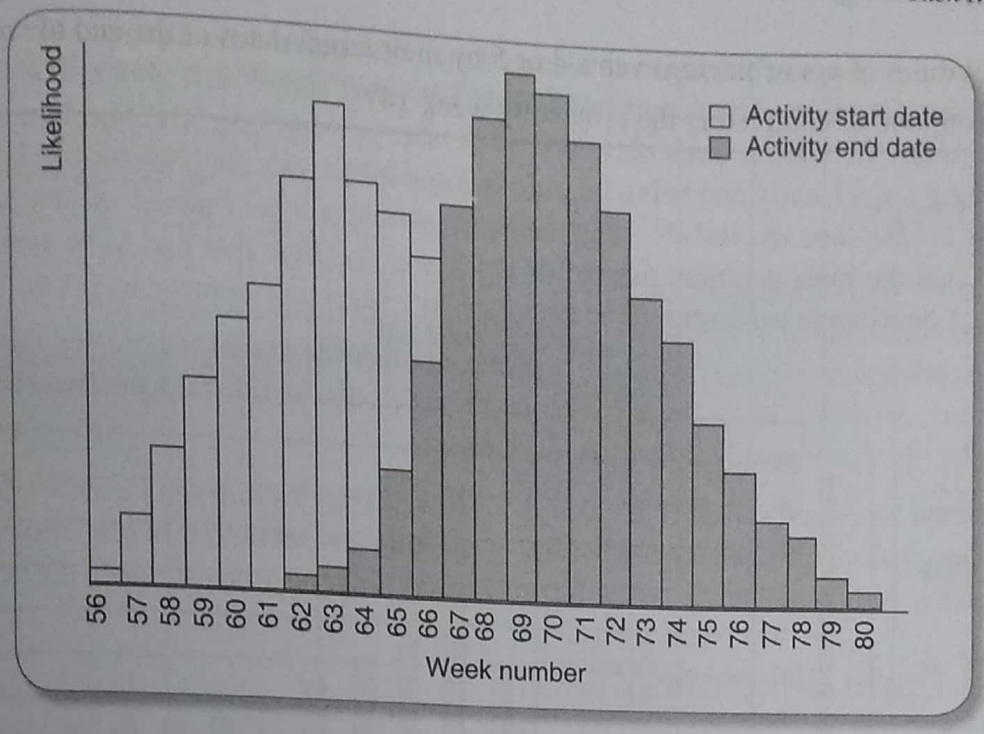


FIGURE 7.9 Risk profile for an activity generated using Monte Carlo simulation

To appreciate the advantage of Monte Carlo simulations over a manual approach, consider the following. In the manual approach, a few combinations of each project duration are chosen (such as best case, worst case, and most likely case), and the results recorded for each selected scenario. In contrast, in Monte Carlo simulation, hundreds or thousands of possible random sampling of probability distribution functions of the activity durations are considered as samples for evaluation of the project completion time expression to produce outcomes. Monte Carlo simulation is expected to give a more realistic result than manual analysis of a few cases, especially because manual analysis implicitly gives equal weights to all scenarios.

7.14 Critical Chain Concepts

This chapter has stressed the idea that the forecast for the duration of an activity cannot in reality be a single number, but must be a range of durations that can be displayed on a graph such as Figure 7.3. However, we would want to pick one value in that range which would be the *target*.

The duration chosen as the target might be the one that seems to be the *most likely*. Imagine someone who cycles to work each day. It may be that on average it takes them about 45 minutes to complete the journey, but on some days it could be more and on others it could be less. These journey times could be plotted on a graph like the one in Figure 7.3. If the cyclist had a very important meeting at work, it is likely that they would give themselves more time – say an extra 15 minutes – than the average 45 minutes to make sure that they arrived in time. In the discussion above on the PERT risk technique the most likely duration was the middle value and the pessimistic estimate was the equivalent of the 45 + 15 = 60 minutes.

Of course, there will be some days when the cyclist will beat the average of 45 minutes. When a project is actually being executed, the project manager will be forced to focus on the activities where the actual durations exceed the target. Activities which are actually completed before the target date are likely to be overlooked. These early completions, properly handled, could put some time in hand that might still allow the project to meet its target completion date if the later activities are delayed.

Figure 7.10 shows the findings of Michiel van Genuchten, a researcher who analysed the reasons for delays in the completion of software development tasks. This bar chart shows that about 30% of activities were finished on time, while 9% were a week early and 17% were a week late. The big jump of 21 percentage points between being a week early and being on time is compatible with the 'Parkinson's Law' principle that '*work expands to fill the time available*'. This tendency should not be blamed on inherent laziness. van Genuchten found that the most common reason for delay was the time that had to be spent on non-project work. It seems that developers used spare time provided by generous estimates to get on with other urgent work.

Michiel van Genuchten (1991) 'Why is software late? An empirical study of reasons for delay in software development', *IEEE Transactions in Software Engineering* 17(6) 582–90.

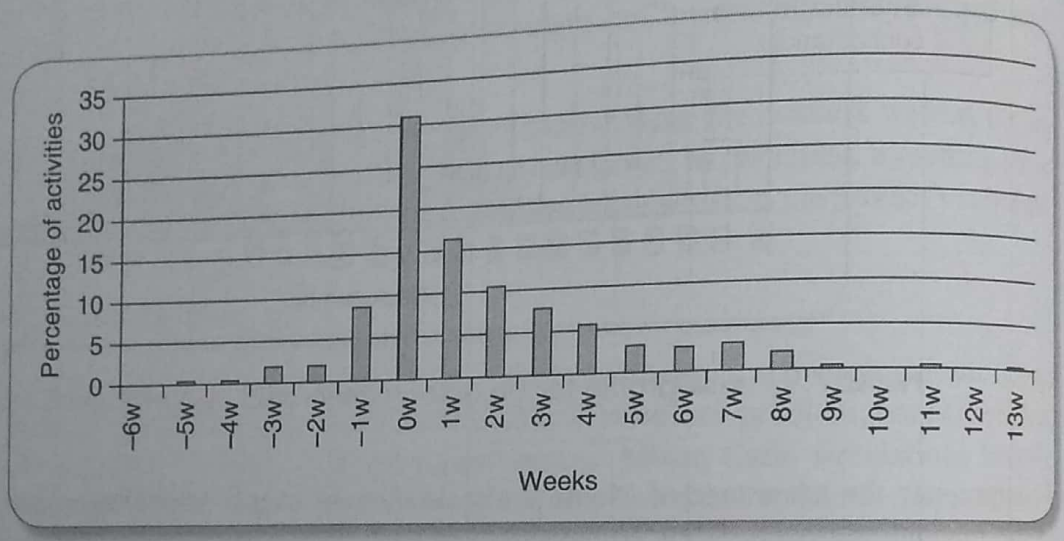


FIGURE 7.10 Percentage of activities early or late (after van Genuchten, 1991)

A good introduction is L. P. Leach (1999) 'Critical chain project management improves project performance' *Project Management Journal* 30(2) 39–51.

One approach which attempts to solve some of these problems is the application of the *critical chain* concept originally developed by Eliyahu Goldratt. In order to demonstrate the principles of this approach, the example shown in Figure 7.7 will be reworked as a Gantt chart. Figure 7.11 shows what the Gantt chart for this project might look like if a 'traditional approach' were adopted, but we have already adopted the most likely durations.

The general steps in the Critical Chain approach are explained in the following sections.

Deriving 'most likely' activity durations

The target date generated by critical chain planning is one where it is estimated that there is a 50% chance of success – this approximates to the expected time identified in the PERT risk method. In some explanations of critical chain project planning it is suggested that the most likely activity duration can be identified by halving the estimates provided. This is based on the assumption that the estimates given to the planner will be 'safe' ones based on a 95% probability of them being achieved. If you look at Figure 7.3, the 95% estimate would be 9 days and half of that (4.5 days) would not be a reasonable target as it would have a probability of only 10% of success. It also assumes that a probability profile has a bell-shaped normal distribution (like the example in Figure 7.3). If you look at the distribution which resulted from van Genuchten's research – see Figure 7.10 – you can see that it is certainly not bell-shaped. Other critical chain experts suggest deducting 33% from the safe estimate to get the target estimate – which seems less unreasonable.

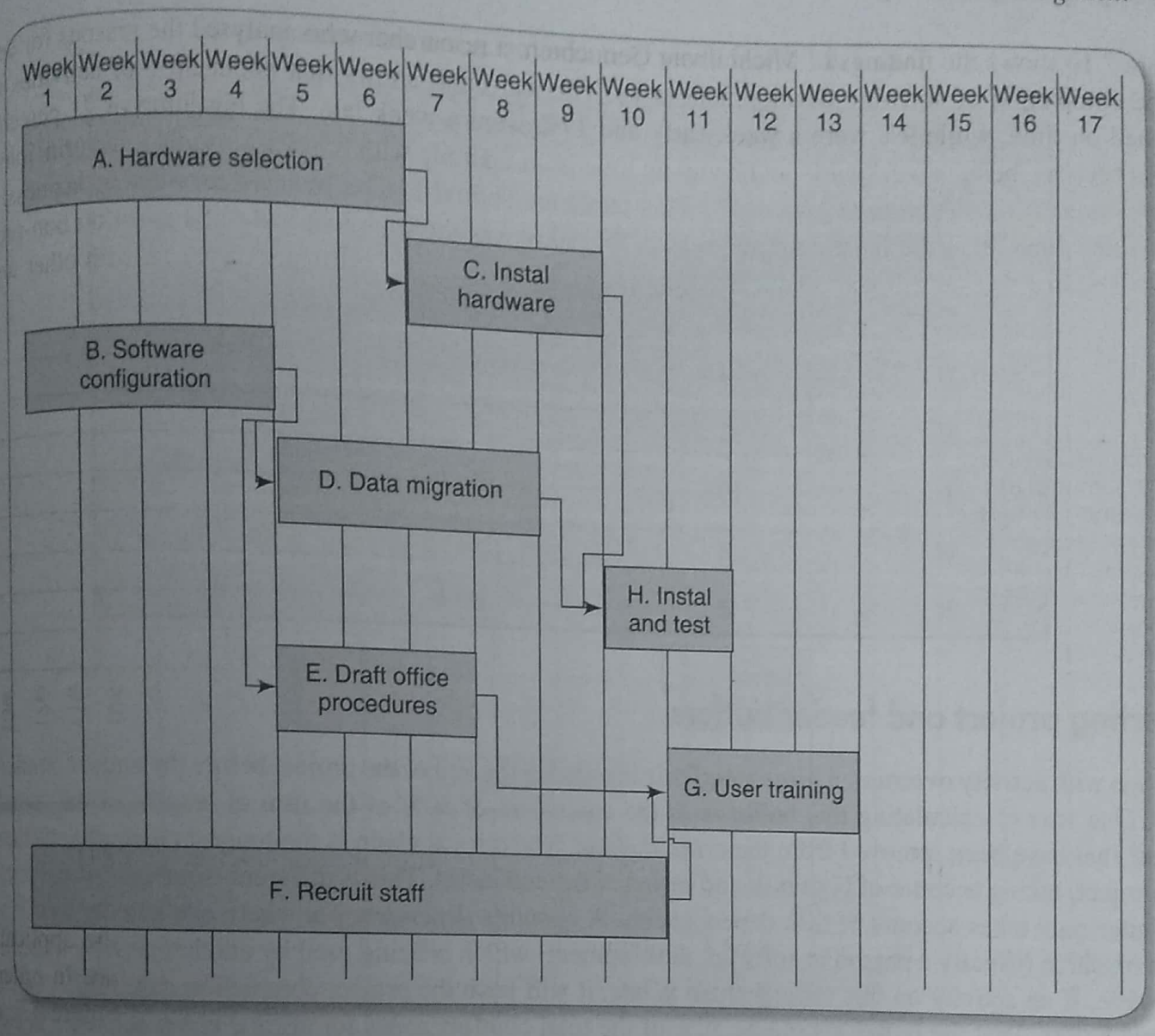


FIGURE 7.11 Gantt chart – 'traditional' planning approach

However, what appear to be arbitrary managerial reductions in the estimates may not be a good way to motivate developers, especially if these staff supplied the estimates in the first place. A better approach would be to ask developers to supply two estimates. One of these would be a 'most likely' estimate and the other would include a safety margin or comfort zone. From now on we are going to assume that this is what has happened. In fact we will use the figures already presented in Table 7.6 in this new role (Table 7.8).

Using latest start dates for activities

Working backwards from the target completion date, each activity is scheduled to start as late as possible. Among other things, this should reduce the chance of staff being pulled off the project on to other work. It is also argued – with some justification according to van Genuchten's research above – that most developers would tend to start work on the task at the latest start time anyway. However, it does make every activity 'critical'. If one is late the whole project is late. That is why the next steps are needed.

TABLE 7.8 Most likely and comfort zone estimates (days)

Activity	Most likely	Plus comfort zone	Comfort zone
A	6	8	2
B	4	5	1
C	3	3	0
D	4	5	1
E	3	4	1
F	10	15	5
G	3	4	1
H	2	2.5	0.5

Inserting project and feeder buffers

To cope with activity overruns, a *project buffer* is inserted at the end of the project before the target completion date. One way of calculating this buffer is as the equivalent of 50% of the sum of lengths of the 'comfort zones' that have been removed from the *critical chain*. The critical chain is the longest chain of activities in the project, taking account of both task and resource dependencies. This is different from the *critical path* as the latter only takes account of task dependencies. A *resource dependency* is where one activity has to wait for a resource (usually a person in software development) which is being used by another activity to become available. If an activity on this critical chain is late, it will push the project completion date further into the project buffer. That the buffer should be 50% of the total comfort zones for critical chain activities is based on the grounds that if the estimate for an activity was calculated as having a 50% chance of being correct, the buffer would only need to be called upon by the 50% of cases where the estimate was not correct.

An alternative proposal is to sum the squares of the comfort zones and then take the square root of the total. This is based on the idea that each comfort zone is the equivalent to the standard deviation of the activity – go back and look at the section headed *Calculating the standard deviation of each project event* in Section 7.12. This method of calculation still produces a figure which is less than simply summing all the comfort zones. This is justified on the grounds that the contingency time needed for a group of activities is less than the sum of the individual contingency allowances as the success of some activities will compensate for the shortfalls in others.

Buffers are also inserted into the project schedule where a subsidiary chain of activities feeds into the critical chain. These *feeding buffers* could once again be set at 50% of the length of the 'comfort zone' removed from the subsidiary or *feeding chain*.

Worked example

Figure 7.12 shows the results of this process. The critical chain in this example happens to be the same as the critical path, that is activities F and G which have comfort zones of 5 weeks and 1 week respectively, making a total of 6 weeks. The project buffer is therefore 3 weeks.

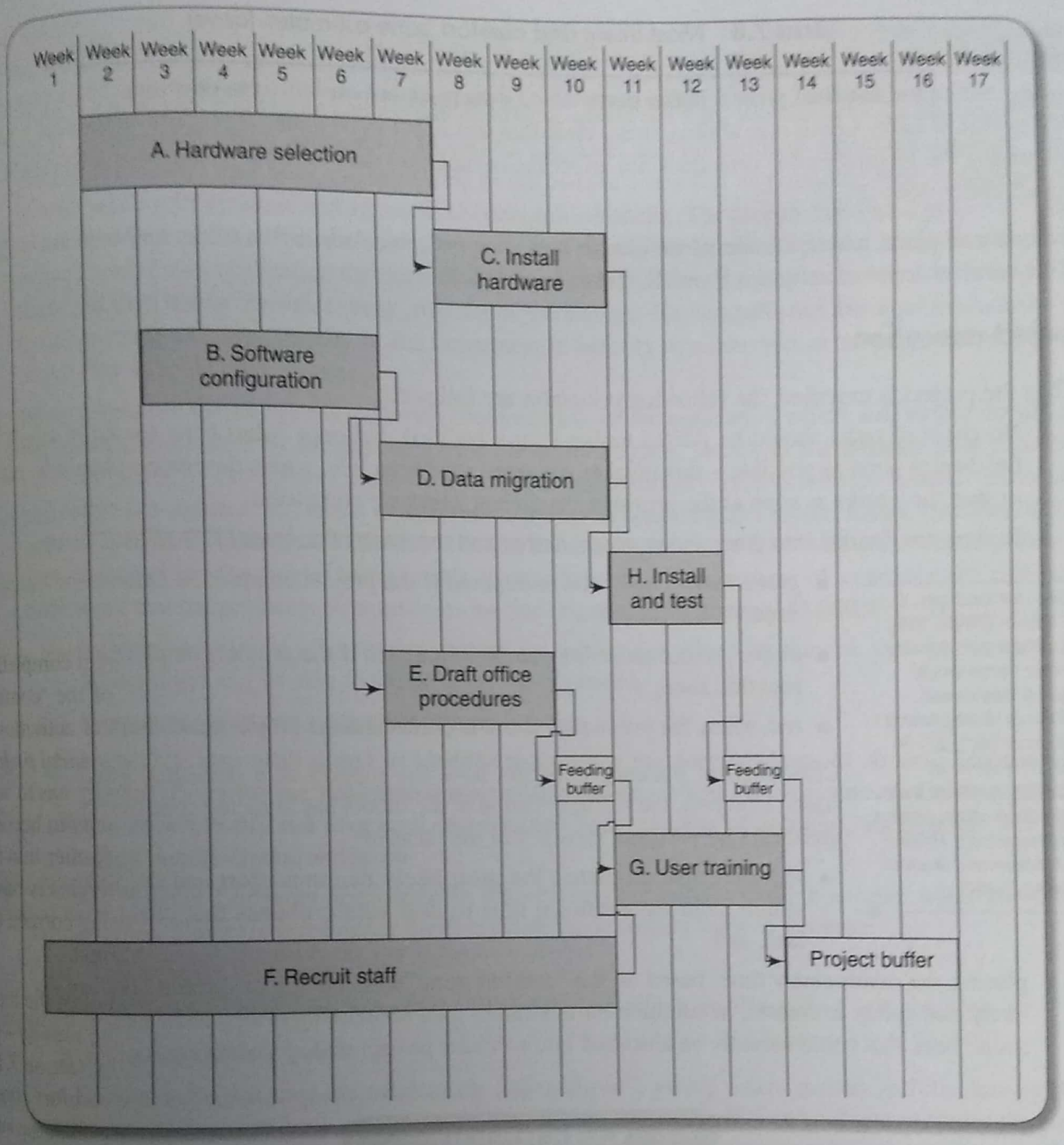


FIGURE 7.12 Gantt chart – critical chain planning approach

Subsidiary chains feed into the critical chain where activity H links into the project buffer and where activity E links into G which is part of the critical chain. Feeding buffers are inserted at these points. For the first buffer the duration would be 50% of the saved comfort zones of A, C and H, that is $(2 + 0 + 0.5)/2 = 1.25$ weeks. It could be argued that B, D and H could form a feeder chain which also has a combined comfort zone of $(1 + 1 + 0.5)/2 = 1.25$ weeks. In the situations where there are parallel alternative paths on a feeder chain, the practice is to base the feeding buffer on whichever comfort zone total is greater. This because if one or other or both parallel paths were late they could still use the same buffer. (Imagine that in the example above there are two cyclists who live 45 minutes away from work and they both have the same important

meeting – they might each add a 15-minute comfort zone to the ride on that day but that 15 minutes could effectively be the same 15 minutes between 7.45 and 8.00 a.m. in the morning). It could be argued that the feeding buffer and the final project buffer could also be merged, but explanations of critical chain planning such as that of Larry Leach (see above), make clear that this is not to be done. This could be because a delay penetrating the feeding buffer time does not affect the completion date of the project, while penetrating the project buffer does.

In the second place, where a feeder chain of activities joins the critical chain, the feeder buffer would be 50% of the comfort zones of activities B and E, that is 1 week.

Project execution

When the project is executed, the following principles are followed:

- No chain of tasks should be started earlier than scheduled, but once it has been started it should be finished as soon as possible – this invokes the *relay race principle*, where developers should be ready to start their tasks as soon as the previous, dependent, tasks are completed.
- Buffers are divided into three zones: green, amber and red, each of an even (33%) size:
  - *green*, where no action is required if the project completion date creeps into this zone;
  - *amber*, where an action plan is formulated if the project completion dates moves into this zone;
  - *red*, where the action plan above is executed if the project penetrates this zone.

Critical chain planning concepts have the support of a dedicated group of enthusiasts. However, the full application of the model has attracted controversy on various grounds. Our personal view is that the ideas of:

- requiring two estimates: the most likely duration/effort and the safety estimate which includes additional time to deal with problems that could arise with the task, and
- placing the contingency time, based on the 'comfort zone' which is the difference between the most likely and safety estimates, in common buffers rather than associating it with individual activities are sound ones that could usefully be absorbed into software project management practice.

See, for example, D. Trietsch (2005) 'Why a critical path by any other name would smell less sweet' *Project Management Journal* 36(1) 27–36 and T. Raz *et al.* (2003) 'A critical look at critical chain project management' *Project Management Journal* 34(4) 24–32.

Conclusion

In this chapter, we have seen how to identify and manage the risks that might affect the success of a project. Risk management is concerned with assessing and prioritizing risks and drawing up plans for addressing those risks before they become problems.

This chapter has also described techniques for estimating the effect of risk on the project's activity network and schedule.

Many of the risks affecting software projects can be reduced by allocating more experienced staff to those activities that are affected. In the next chapter we consider the allocation of staff to activities in more detail.