9 Monitoring and Control

Learning Objectives

- Monitor the progress of projects
- Assess the risk of slippage
- Visualize and assess the state of a project
- Revise targets to correct or counteract drift
- Control changes to a project's requirements

9.1 Introduction

Once work schedules have been published and the project is started, attention must be focused on progress. This requires monitoring of what is happening, comparison of actual achievement against the schedule and, where necessary, revision of plans and schedules to bring the project as far as possible back on target.

In earlier chapters we have stressed the importance of producing plans that can be monitored – for example, ensuring that activities have clearly defined and visible completion points. We will discuss how information about project progress is gathered and what actions must be taken to ensure that a project meets its targets.

The final part of this chapter discusses how we can deal with changes that are imposed from outside – in particular, changes in requirements.

9.2 Creating the Framework

Exercising control over a project and ensuring that targets are met is a matter of regular monitoring – finding out what is happening and comparing it with targets. There may be a mismatch between the planned outcomes and the actual ones. Replanning may then be needed to bring the project back on target. Alternatively, the target could have to be revised. Figure 9.1 illustrates a model of the project control cycle and shows how, once the initial project plan has been published, project control is a continual process of monitoring progress against that plan and, where necessary, revising the plan to take account of deviations. It also illustrates the important steps that must be taken after completion of the project so that the experience gained in any one project can feed into the planning stages of future projects, thus allowing us to learn from past mistakes.

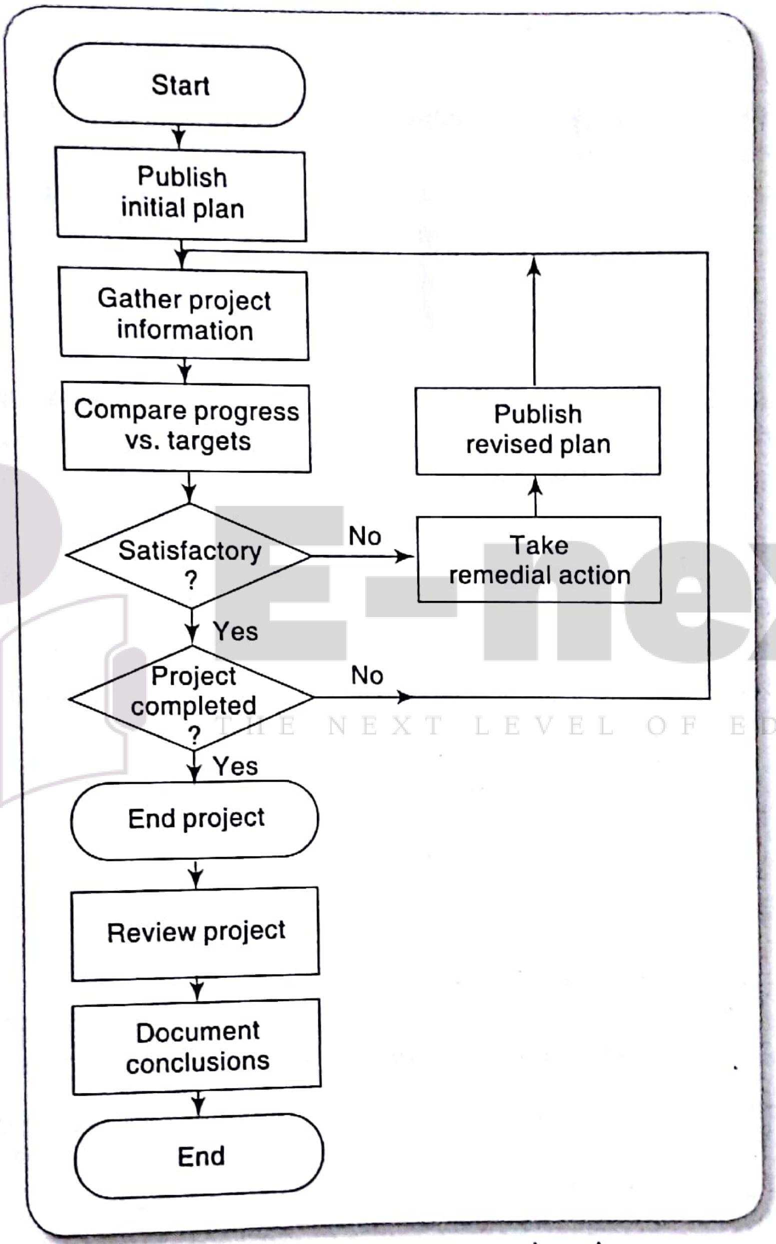

FIGURE 9.1 Project control cycle

In practice we are normally concerned with four types of shortfall – delays in meeting target dates, shortfalls in quality, inadequate functionality, and costs going over target. In this chapter we are mainly concerned with the first and last of these.

See Chapter 13 for a discussion on software quality.

Responsibility

The overall responsibility for ensuring satisfactory progress on a project is often the role of the *project steering committee*, *project management board* or, in PRINCE2, *Project Board*. Day-to-day responsibility will rest with the project manager and, in all but the smallest of projects, aspects of this can be delegated to team leaders.

The concept of a reporting hierarchy was introduced in Chapter 1.

Figure 9.2 illustrates the typical reporting structure found with medium and large projects. With small projects (employing around half a dozen or fewer staff) individual team members usually report directly to the project manager, but in most cases team leaders will collate reports on their section's progress and forward summaries to the project manager. These, in turn, will be incorporated into project-level reports for the steering committee and, via them or directly, progress reports for the client.

In a PRINCE2 environment, there may be a Project Assurance function reporting to the Project Board and independent of the Project Manager.

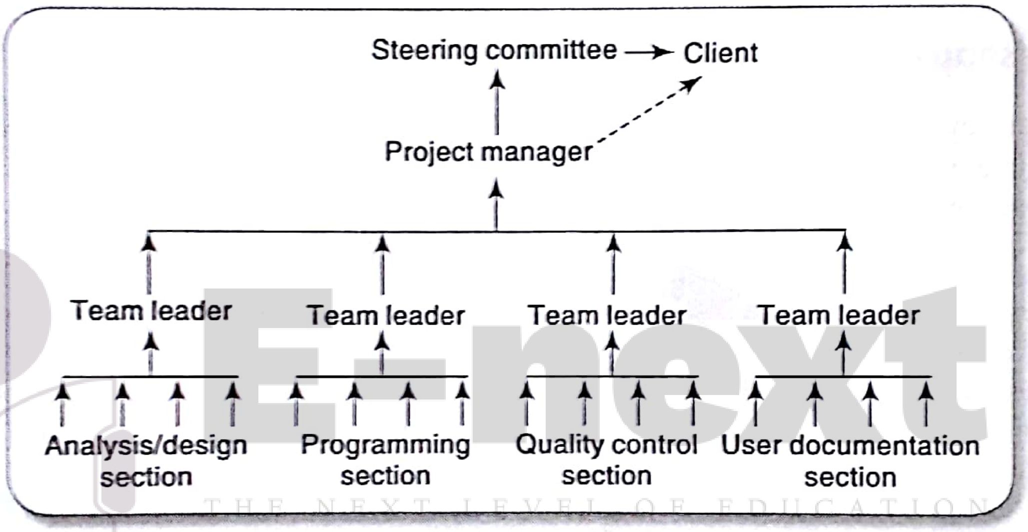

Figure 9.2 Project reporting structures

Chapter 12 will explore communication in a more general project context.

Reporting may be oral or written, formal or informal, and regular or ad hoc – see Table 9.1. Informal communication is necessary and important, but any such informal reporting of project progress must be complemented by formal reporting procedures – and it is those we are concerned with in this chapter.

Table 9.1 Categories of reporting

Report type	Examples	Comment
Oral formal regular	Weekly or monthly progress meetings	While reports may be oral, formal written minutes should be kept
Oral formal ad hoc	End-of-stage review meetings	While largely oral, likely to receive and generate written reports
Written formal regular	Job sheets, progress reports	Normally weekly using forms
Written formal ad hoc	Exception reports, change reports	
Oral informal ad hoc	Canteen discussion, social interaction	Often provides early warning; must be backed up by formal reporting

Assessing progress

Some information used to assess project progress will be collected routinely, while other information will be triggered by specific events. Wherever possible, this information should be objective and tangible – whether or not a particular report has been delivered, for example. Sometimes, however, assessment will have to depend on estimates of the proportion of the current activity that has been completed.

Setting checkpoints

It is essential to set a series of checkpoints in the initial activity plan. Checkpoints may be:

- Regular (monthly, for example)
- Tied to specific events such as the production of a report or other deliverable

The PRINCE2 standard described in Appendix A has its own terminology.

Taking snapshots

The frequency of progress reports will depend upon the size and degree of risk of the project. Team leaders, for example, may want to assess progress daily (particularly when employing inexperienced staff) whereas project managers may find weekly or monthly reporting appropriate. In general, the higher the level, the less frequent and less detailed the reporting needs to be.

At the level of individual developers, however, strong arguments exist for the formal weekly collection of information. This ensures that information is provided while memories are still relatively fresh and provides a mechanism for individuals to review and reflect upon their progress. If reporting is to be weekly then it makes sense to have basic units of work that last about a week.

Recall that in Chapter 4, Beck recommended weekly work cycles in an XP environment.

Major, or project-level, progress reviews will generally take place at particular points during the life of a project – commonly known as *review points* or *control points*. PRINCE2, for example, designates a series of checkpoints where the status of work in a project or for a team is reviewed. At the end of each project Stage, PRINCE2 provides for an End Stage Assessment where an assessment of the project and consideration of its future are undertaken.

9.3 Collecting the Data

As a rule, managers will try to break down long activities into more controllable tasks of one or two weeks' duration. However, it will still be necessary to gather information about partially completed activities and, in particular, forecasts of how much work is left to be completed. It can be difficult to make such forecasts accurately.

Exercise 9.1

A software developer working on Amanda's project has written the first 250 lines of a Java program that is estimated to require 500 lines of code. Explain why it would be unreasonable to assume that the programming task is 50% complete.

How might you make a reasonable estimate of how near completion it might be?

Where there is a series of products, partial completion of activities is easier to estimate. Counting the number of record specifications or screen layouts produced, for example, can provide a reasonable measure of progress.

In some cases, intermediate products can be used as in-activity milestones. The first successful compilation of a program, for example, might be considered a milestone even though it is not a final product.

Partial completion reporting

Weekly timesheets are a valuable source of information about resources used. They are often used to provide information about what has been achieved. However, requesting partial completion estimates where they cannot be obtained from objective measures encourages the 99%

Many organizations use standard accounting systems with weekly timesheets to charge staff time to individual jobs. The staff time booked to a project indicates the work carried out and the charges to the project. It does not, however, tell the project manager what has been produced or whether tasks are on schedule.

It is therefore common to adapt or enhance existing accounting data collection systems to meet the needs of project control. Weekly timesheets, for example, are frequently adapted by breaking jobs down to activity level and requiring information about work done in addition to time spent. Figure 9.3 shows an example of a report form, in this case requesting information about likely slippage of completion dates as well as estimates of completeness. Other reporting templates are possible. For example,

Time Sheet

Staff John Smith Week ending 30/3/07

Rechargeable hours

Project	Activity code	Description	Hours this week	% complete	Scheduled completion	Estimated completion
P21	A243	Code mod A3	12	30	24/4/07	24/4/07
P34	B771	Document take-on	20	90	6/4/07	4/4/07

Total recharged hours	32

Non-rechargeable hours

Code	Description	Hours this week	Comment and authorization
Z99	Day in lieu	8	Authorized by RB

Total non-rechargeable hours	8

Figure 9.3 A weekly timesheet and progress review form

rather than ask for estimates of percentage complete, some managers would prefer to ask for the number of hours already worked on the task and an estimate of the number of hours needed to finish the task off.

Red/amber/green (RAG) reporting

One popular way of overcoming the objections to partial completion reporting is to avoid asking for estimated completion dates, but to ask instead for the team members' estimates of the likelihood of meeting the planned target date.

Complete syndrome – tasks are reported as on time until 99% complete, and then stay at 99% complete until finished.

One way of doing this is the traffic-light method. This consists of the following steps:

- Identify the key (first level) elements for assessment in a piece of work
- Break these key elements into constituent elements (second level)
- Assess each of the second-level elements on the scale *green* for 'on target', *amber* for 'not on target but recoverable', and *red* for 'not on target and recoverable only with difficulty'
- Review all the second-level assessments to arrive at first-level assessments
- Review first- and second-level assessments to produce an overall assessment

There are a number of variations on the traffic-light technique. The version described here is in use in IBM and is described in A. Down, M. Coleman and P. Absolon (1994) *Risk Management for Software Projects*, McGraw-Hill.

For example, Amanda decides to use a version of the traffic-light method for reviewing activities on the IOE project. She breaks each activity into a number of component parts (deciding, in this case, that a further breakdown is unnecessary) and gets the team members to complete a return at the end of each week. Figure 9.4 illustrates Justin's completed assessment at the end of week 16.

Activity Assessment Sheet

Staff Justin

Ref: IoE/P/13 Activity: Code and test module C

Week number	13	14	15	16	17	18	
Activity summary	G	A	A	R			
Component							Comments
Screen handling procedures	G	A	A	G			
File update procedures	G	G	R	A			
Housekeeping procedures	G	G	G	A			
Compilation	G	G	G	R			
Test data runs	G	G	G	A			
Program documentation	G	G	A	R			

FIGURE 9.4 A traffic-light assessment of IoE/P/13

Note that this form refers only to uncompleted activities. Justin would still need to report activity completions and the time spent on activities.

Traffic-light assessment highlights only risk of non-achievement; it is not an attempt to estimate work done or to quantify expected delays.

Following completion of assessment forms for all activities, the project manager uses these as a basis for evaluating the overall status of the project. Any critical activity classified as amber or red will require further consideration and often leads to a revision of the project schedule. Non-critical activities are likely to be considered as a problem if they are classified as red, especially if all their float is likely to be consumed.

9.4 Review

From a manager's perspective, review of work products is an important mechanism for monitoring the progress of a project and ensuring the quality of the work products.

Every project is developed through iterations over a large number of work products such as requirements document, design document, project plan document, code, etc. Each of these work products can have a large number of defects in them due to mistakes committed by the development team members. It is necessary to eliminate as many defects in these work products to realize a product of acceptable quality. Testing is an effective defect removal mechanism. However, testing is applicable to only executable code. How can the defects from the non-executable work products such as requirements document and design document be removed? Review is a very effective technique to remove defects from all work products including code. In fact, review has been acknowledged to be more cost-effective in removing defects as compared to testing. Early review techniques focused on code and systematic review techniques were developed for this specific purpose. But over the years, review techniques have become extremely popular and have been generalized for use with other work products.

Utility of review

Besides being a cost-effective defect removal mechanism, review of any work product has several other benefits including the following mentioned below:

- Review usually helps to identify any deviation from standards, including issues that might affect maintenance of the software.
- Reviewers suggest ways to improve the work product such as using algorithms that are more time or space efficient, specific work simplifications, better technology opportunities that can be exploited, etc.
- In addition to defect identification, a review meeting often provides learning opportunities to not only the author of a work product, but also the other participants of the review meeting. The lessons acquired from a review meeting allows participants to avoid committing similar defects that were discussed in the review meeting and also allows them to make use of the best practices that were suggested.
- The review participants gain a good understanding of the work product under review, making it easier for them to interface or use the work product in their work.

Exercise 9.2

For removing bugs from code, would review or testing be more cost-effective? Explain the reason behind your answer.

Candidate work products for review

All interim and final work products are usually candidates for review. Usually, the work products considered to be suitable candidates for review are as follows:

- Requirements specification documents
- User interface specification and design documents
- Architectural, high-level, and detailed design documents
- Test plan and the designed test cases
- Project management plan and configuration management plan

Review roles

In every review meeting, a few key roles need to be assigned to the review team members. These roles are moderator, recorder and the reviewers. The moderator plays a key role in the review process. The principal responsibilities of the moderator include scheduling and convening meetings, distributing review materials, leading and moderating the review sessions, ensuring that the defects are tracked to closure. The main role of the recorder is to record the defects found, the time, and effort data. The review team members review the work product and give specific suggestions to the author about the existing defects and also point out ways to improve the work product.

In the following, we will first discuss a generic review process. Subsequently, we shall discuss the importance of collection of all relevant data to the success of the review process.

Review process

Review of any work product consists of the following four important activities, viz. planning, review preparation and overview, review meeting, rework and follow-up. A review process model is shown in Figure 9.5. The model in Figure 9.5 captures the sequence of the activities that need to be carried out, the input to the activities, and the output produced from the activities. In the following, we briefly discuss these review activities.

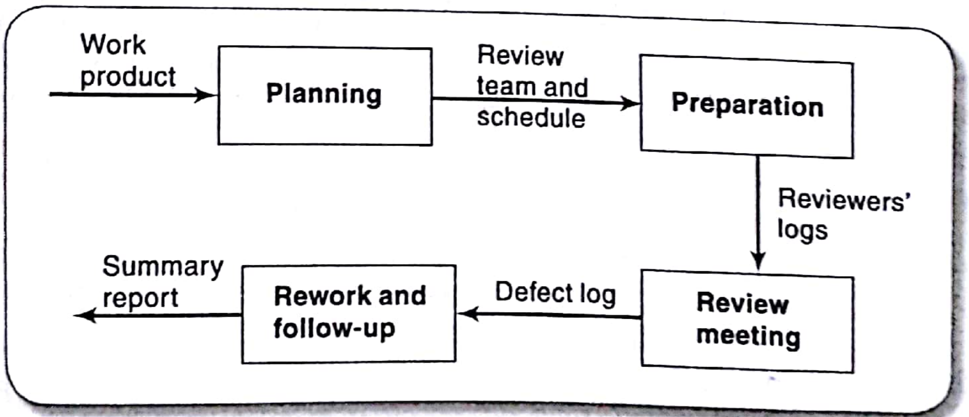

FIGURE 9.5 Review process model

- *Planning* Once the author of a work product is ready for submitting the work for review; the project manager nominates a moderator. A moderator can be someone who is familiar with the work product. In consultation with the moderator, the project manager nominates the other members of the review team. Usually, the review process works best when the number of members is between five and seven.

The effectiveness of review drastically reduces if there are less than three members. The review team is usually selected from the following types of project team members.

- The author of the preceding work product based on which the work product under review was developed
- The member who would use the work product under review
- Peers of the author
- The authors of the work products that would interface with the work product under review

The moderator usually schedules all review meetings.

- *Preparation* To initiate the review process, the moderator convenes a brief preparation meeting. In the preparation meeting, copies of the work product are distributed to the review team members. The author presents a brief overview of the work product. The moderator highlights the objectives of the review. The reviewers then individually carry out review and record their observations in separate documents called review logs.
- *Review Meeting* In the review meeting the reviewer's give their comments based on the logs they have prepared beforehand. The comments may pertain to a defect, work simplification, maintainability, etc. The author responds to the reviewers' comments and in this discussion other reviewers may also take part. The moderator ensures that the discussions remain focused and productive. The recorder scribes all the defects and points that the author agrees to, as well as the review statistics in the form of a review log.
- *Rework* The author addresses all the issues raised by the reviewers by carrying out the necessary modifications to the work product and prepares a rejoinder to all the points scribed in the review log. The rejoinder records the exact ways in which the comments have been taken care of by the author. The corrected work product along with the author's rejoinder is circulated among all the review team members. In a final brief meeting, the review team members check whether all the issues scribed in the review log have been resolved satisfactorily. At the end of this meeting, a final summary report of the review is prepared.

Data collection

Since a review meeting is a completely human endeavour, unless the data representing the results of the meetings is properly recorded, it can get lost. In addition to recording all defects, the data about the time spent by the reviewers in the review activity must also be captured. A record of the defect data is needed for tracking defects in the project.

The different reports in which the review data are captured are as follows:

1. *Review Preparation Log* Each reviewer prepares a review preparation log. The different items recorded in it by the reviewer are the data about defects he observes, their locations, their criticality, and the total time spent in doing the review of the work product.
2. *Review Log* In the review log only those defects that are agreed to by the author are logged. Defect logs are a crucial record since these help in tracking all defects to closure.
3. *Review Summary Report* This report summarizes the review data and presents an overall picture of the review. It contains information regarding the total defects and the amount of time spent on each of the review process activities.

9.5 Visualizing Progress

Having collected data about project progress, a manager needs some way of presenting that data to greatest effect. In this section, we look at some methods of presenting a picture of the project and its future. Some of these methods (such as Gantt charts) provide a static picture, a single snapshot, whereas others (such as timeline charts) try to show how the project has progressed and changed through time.

Gantt chart

One of the simplest and oldest techniques for tracking project progress is the Gantt chart. This is essentially an activity bar chart indicating scheduled activity dates and durations, frequently augmented with activity floats. Reported progress is recorded on the chart (normally by shading activity bars) and a 'today cursor' provides an immediate visual indication of which activities are ahead or behind schedule. Figure 9.6 shows part of Amanda's Gantt chart as at the end of Tuesday of week 17. 'Code and test module D' has been completed ahead of schedule and 'Code and test module A' appears also to be ahead of schedule. The coding and testing of the other two modules are behind schedule.

Henry Gantt (1861–1919) was an industrial engineer interested in the efficient organization of work.

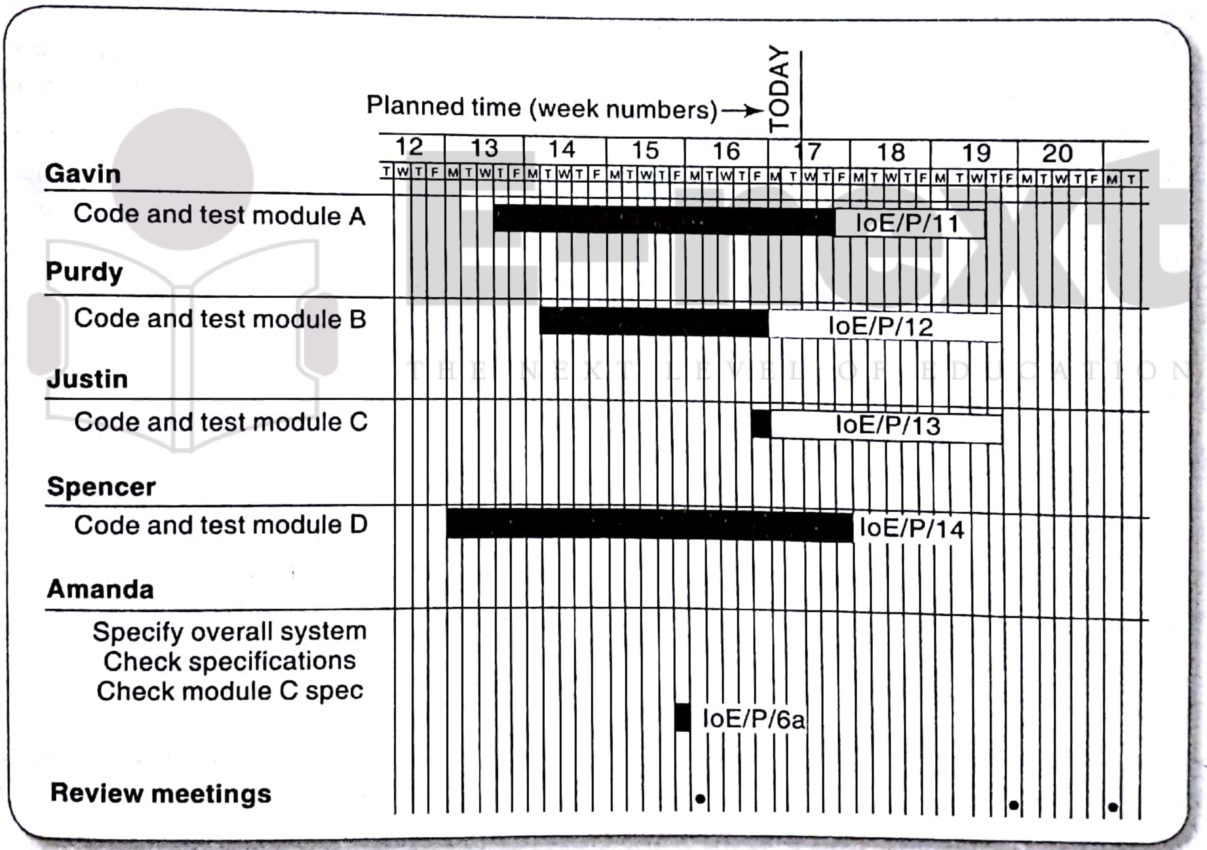

FIGURE 9.6 Part of Amanda's Gantt chart with the 'today cursor' in week 17

Slip chart

A slip chart (Figure 9.7) is a very similar alternative favoured by some project managers who believe it provides a more striking visual indication of those activities that are not progressing to schedule – the more

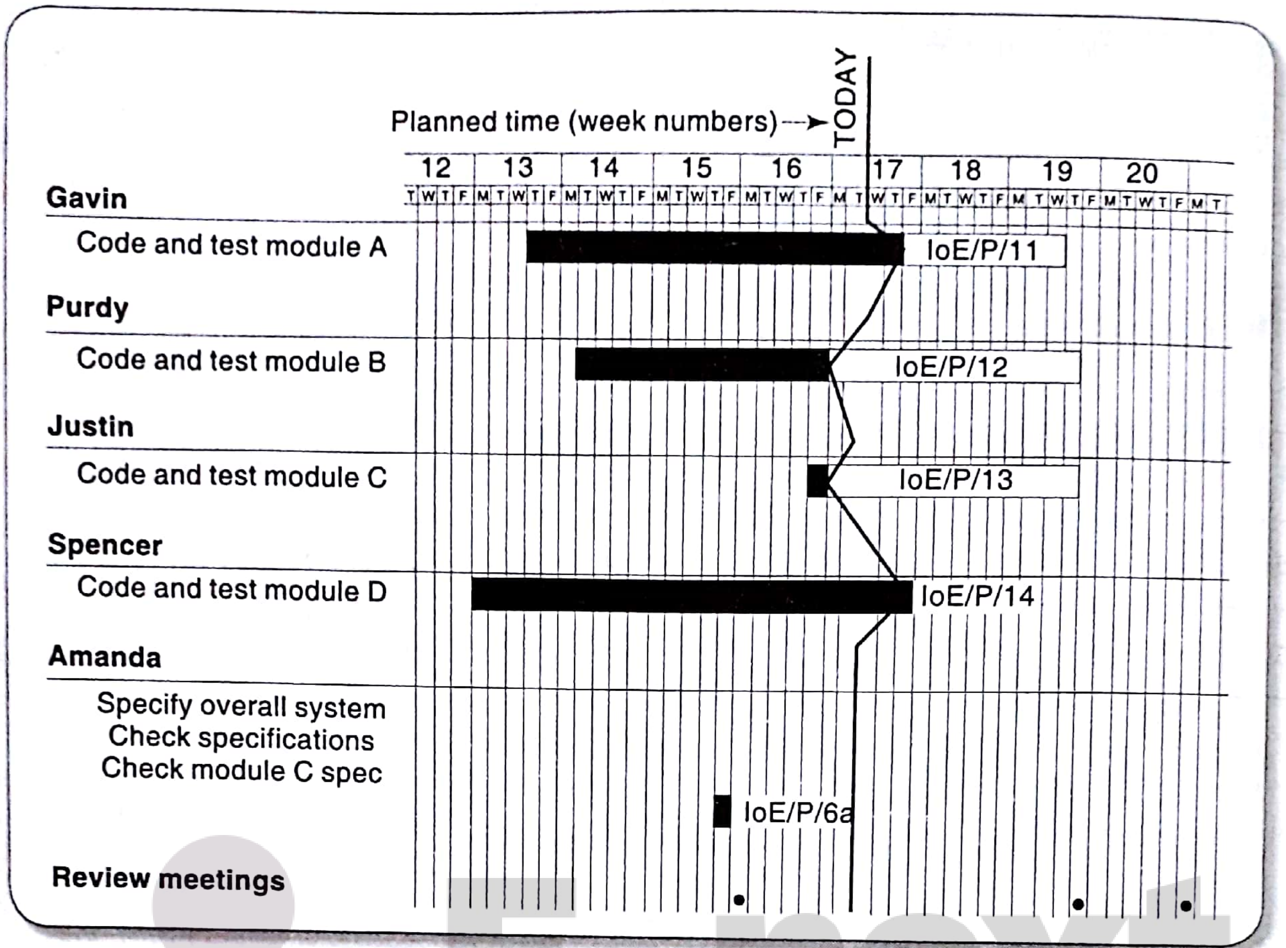

FIGURE 9.7 The slip chart emphasizes the relative position of each activity

the slip line bends, the greater the variation from the plan. Additional slip lines are added at intervals and, as they build up, the project manager will gain an idea as to whether the project is improving (subsequent slip lines bend less) or not. A very jagged slip line indicates a need for rescheduling.

Timeline

One disadvantage of the charts described so far is that they do not show clearly the slippage of the project completion date through the life of the project. Analysing and understanding trends in the project so far allows us to predict the future progress of the project. For example, if a project is behind schedule because so far productivity has not been as high as assumed at the planning stage, it is likely that the scheduled completion date will be pushed back even further unless action is taken to compensate for or improve productivity.

The timeline chart is a method of recording and displaying the way in which targets have changed throughout the duration of the project.

Figure 9.8 shows a timeline chart for Brigette's project at the end of the sixth week. Planned time is plotted along the horizontal axis and elapsed time down the vertical axis. The lines meandering down the chart represent scheduled activity completion dates – at the start of the project 'analyse existing system' is scheduled to be completed by the Tuesday of week 3, 'obtain user requirements' by Thursday of week 5, 'issue tender', the final activity, by Tuesday of week 9, and so on.

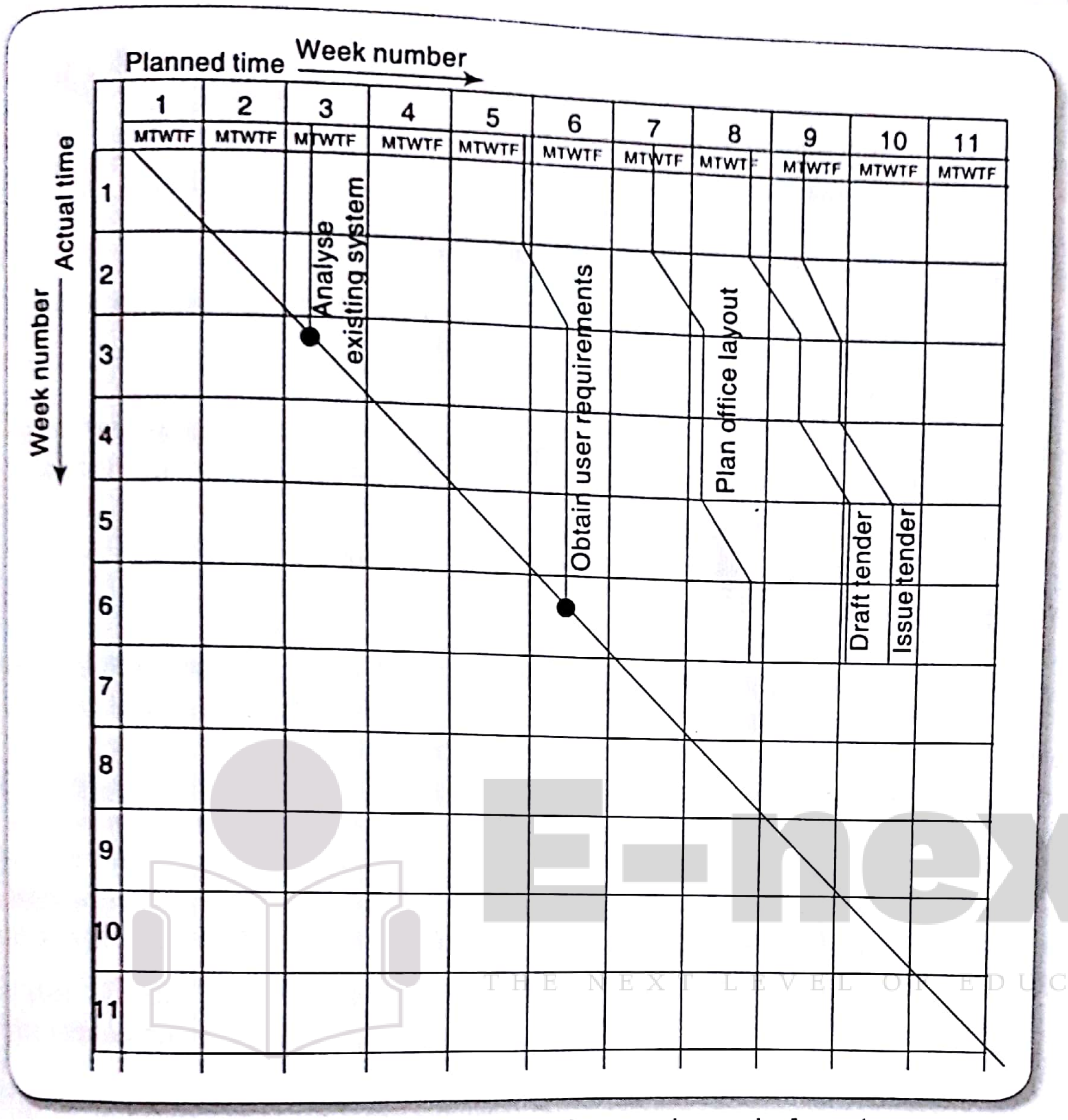

FIGURE 9.8 Brigette's timeline chart at the end of week six

Brigette's timeline chart contains only the critical activities for her project; • indicates actual completion of an activity.

For the sake of clarity, the number of activities on a timeline chart must be limited. Using colour helps to distinguish activities, particularly where lines cross

At the end of the first week Brigette reviews these target dates and leaves them as they are – lines are therefore drawn vertically downwards from the target dates to the end of week 1 on the actual time axis.

At the end of week 2, Brigette decides that 'obtain user requirements' will not be completed until Tuesday of week 6 – she therefore extends that activity line diagonally to reflect this. The other activity completion targets are also delayed correspondingly.

By the Tuesday of week 3, 'analyse existing system' is completed and Brigette puts a blob on the diagonal timeline to indicate that this has happened. At the end of week 3 she decides to keep to the existing targets.

At the end of week 4 she adds another three days to 'draft tender' and 'issue tender'.

Note that, by the end of week 6, two activities have been completed and three are still unfinished. Up to this point she has revised target dates on three occasions and the project as a whole is running seven days late.

Exercise 9.3

By the end of week 8 Brigette has completed planning the office layout but finds that drafting the tender is going to take one week longer that originally anticipated.

What will Brigette's timeline chart look like at the end of week 8?

If the rest of the project goes according to plan, what will Brigette's timeline chart look like when the project is completed?

The timeline chart is useful both during the execution of a project and as part of the post-implementation review. Analysis of the timeline chart, and the reasons for the changes, can indicate failures in the estimation process or other errors that might, with that knowledge, be avoided in future.

9.6 Cost Monitoring

Project costs may be monitored by a company's accounting system. By themselves, they provide little information about project status.

Expenditure monitoring is an important component of project control, not only in itself, but also because it provides an indication of the effort that has gone into (or at least been charged to) a project. A project might be on time but only because more money has been spent on activities than originally budgeted. A cumulative expenditure chart such as that shown in Figure 9.9 provides a simple method of comparing actual and planned expenditure. By itself it is not particularly meaningful – Figure 9.9 could, for example, illustrate a project that is running late or one that is on time but has shown substantial costs savings. We need to take account of the current status of the project activities before attempting to interpret the meaning of recorded expenditure.

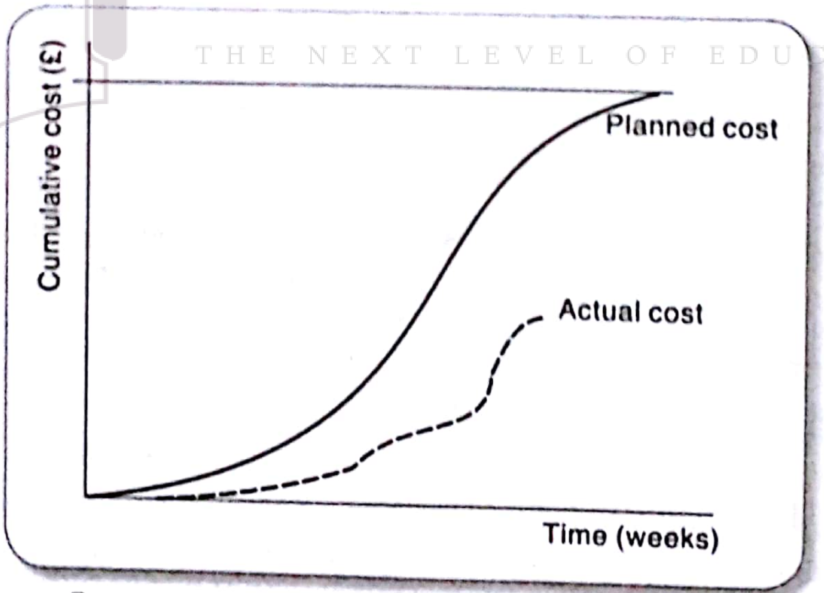

FIGURE 9.9 Tracking cumulative expenditure

Project costs augmented by project monitoring can be used to generate forecasts of future costs.

Cost charts become much more useful if we add projected future costs calculated by adding the estimated costs of uncompleted work to the costs already incurred. Where a computer-based planning tool is used, revision of cost schedules is generally provided automatically once actual expenditure has been recorded. Figure 9.10 illustrates the additional information available once the revised cost schedule is included – in this case it is apparent that the project is behind schedule and over budget.

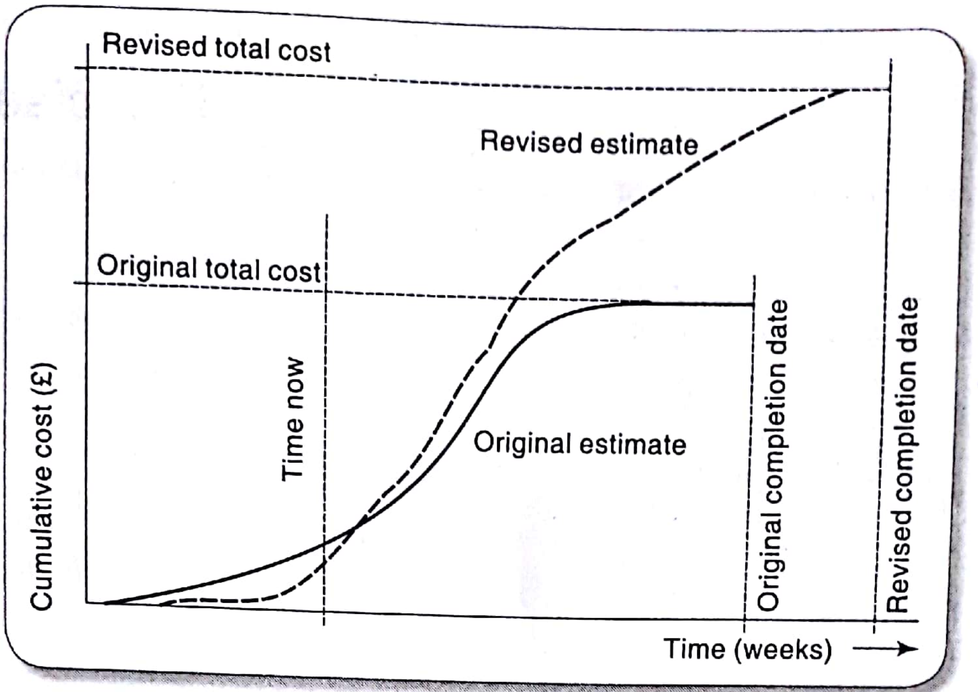

Figure 9.10 The cumulative expenditure chart can also show revised estimates of cost and completion date

9.7 Earned Value Analysis

Earned value analysis has gained in popularity in recent years and may be seen as a refinement of the cost monitoring discussed in the previous section. It originated in the USA's Department of Defence (DOD) as a part of a set of measures to control projects being carried out by contractors for the DOD. Earned value analysis is based on assigning a 'value' to each task or work package (as identified in the WBS) based on the original expenditure forecasts. One way of looking at this is as the equivalent of the price that might be agreed by a contractor to do the unit of work. The assigned value is the original budgeted cost for the item and is known as the *planned value* (PV) or *budgeted cost of work scheduled* (BCWS). A task that has not started is assigned an earned value of zero and when it has been completed, it, and hence the project, is credited with the original planned value of the task. The total value credited to a project at any point is known as the *earned value* (EV) or budgetted cost of work performed (BCWP) and this can be represented as a money value, an amount of staff time or as a percentage of the PV. EV is thus analogous to the agreed price to be paid to the contractor once the work is completed.

> Earned value analysis, also known as budgeted cost of work performed, is recommended by a number of agencies including the US and Australian departments of defence. It is also recommended in BS 6079.

Where tasks have been started but are not yet complete, some consistent method of assigning an earned value must be applied. Common methods in software projects are:

- *the 0/100 technique:* where a task is assigned a value of zero until such time that it is completed when it is given a value of 100% of the budgeted value;
- *the 50/50 technique:* where a task is assigned a value of 50% of its value as soon as it is started and then given a value of 100% once it is complete – this matches some contractual arrangements where a contractor is given half the agreed price when starting the work, perhaps to help pay for raw materials, and the remainder on successful completion;

- *the 75/25 technique:* where the task is assigned 75% on starting and 25% on completion – this is often used when a large item of equipment is being bought: 75% is paid when the equipment is actually delivered and the remainder when installation and testing has been satisfactorily completed;
- *the milestone technique:* where a task is given a value based on the achievement of milestones that have been assigned values as part of the original budget plan;
- *percentage complete:* in some cases there may be a way of objectively measuring the amount of work completed – for example, as part of the implementation of an information system, a number of data records have to be manually typed into a database and the actual number so far completed can be objectively counted.

Of these, we prefer the 0/100 technique for software development. The 50/50 technique can give a false sense of security by over-valuing the reporting of activity starts. The milestone technique might be appropriate for activities with a long duration estimate but, in such cases, it is better to break that activity into a number of smaller ones.

Baseline budget

The first stage in setting up an earned value analysis is to create the *baseline budget*. The baseline budget is based on the project plan and shows the forecast growth in earned value through time. Earned value may be measured in monetary values but, in the case of staff-intensive projects such as software development, it is common to measure earned value in person-hours or workdays. Amanda's baseline budget, based on the schedule shown in Figure 8.7, is shown in Table 9.2 and diagrammatically in Figure 9.11. Notice that she has based her baseline budget on workdays and is using the 0/100 technique for crediting earned value to the project.

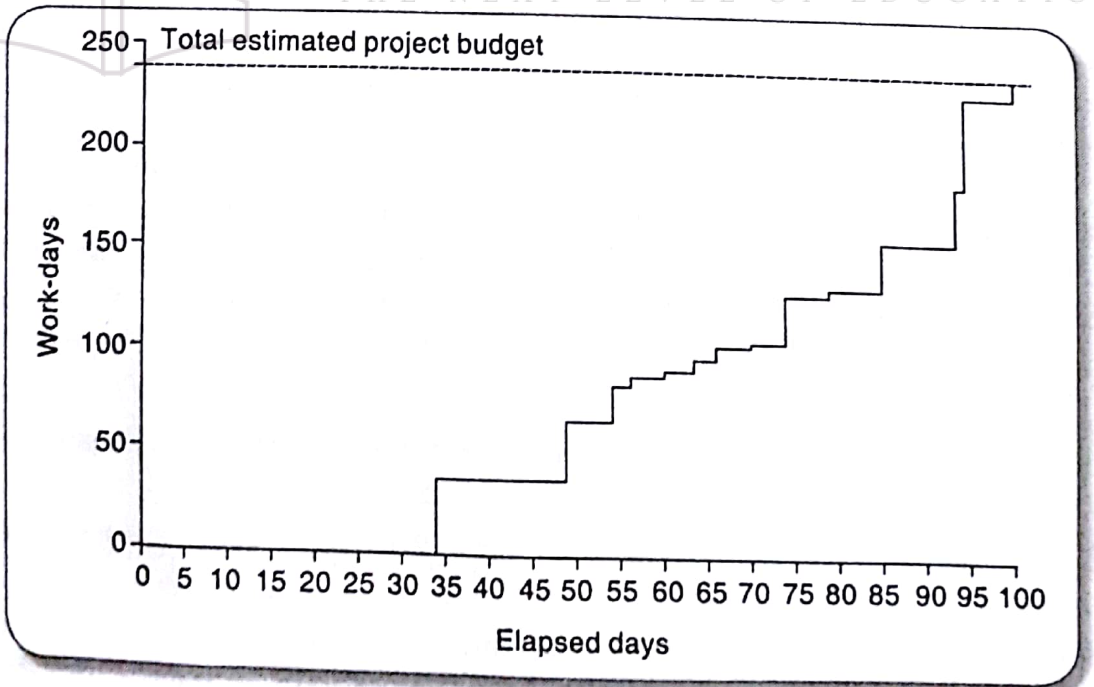

FIGURE 9.11 Amanda's baseline budget

TABLE 9.2 Amanda's baseline budget calculation

Task	Budgeted workdays	Scheduled completion	Cumulative workdays	% cumulative earned value
Specify overall system	34	34	34	14.35
Specify module B	15	49	64	27.00
Specify module D	15	49		
Specify module A	20	54	84	35.44
Check specifications	2	56	86	36.28
Design module D	4	60	90	37.97
Design module A	7	63	97	40.93
Design module B	6	66	103	43.46
Specify module C	25	74	128	54.01
Check module C spec	1	75	129	54.43
Design module C	4	79	133	56.12
Code and test module D	25	85	158	66.67
Code and test module A	30	93	188	79.32
Code and test module B	28	94	231	97.47
Code and test module C	15	94		
System integration	6	100	237	100.00

Amanda's project is not expected to be credited with any earned value until day 34, when the activity 'specify overall system' is to be completed. This activity was forecast to consume 34 person-days and it will therefore be credited with 34 person-days of earned value when it has been completed. The other steps in the baseline budget chart coincide with the scheduled completion dates of other activities.

Monitoring earned value

Having created the baseline budget, the next task is to monitor earned value as the project progresses. This is done by monitoring the completion of tasks (or activity starts and milestone achievements in the case of the other crediting techniques).

Exercise 9.4

Figure 9.12 shows Amanda's earned value analysis at the start of week 12 of the project. Note that here both PV and EV are measured in 'work-days' and that the 0/100 rule is being applied. The earned value (EV) is clearly lagging behind the baseline budget, indicating that the project is behind schedule.

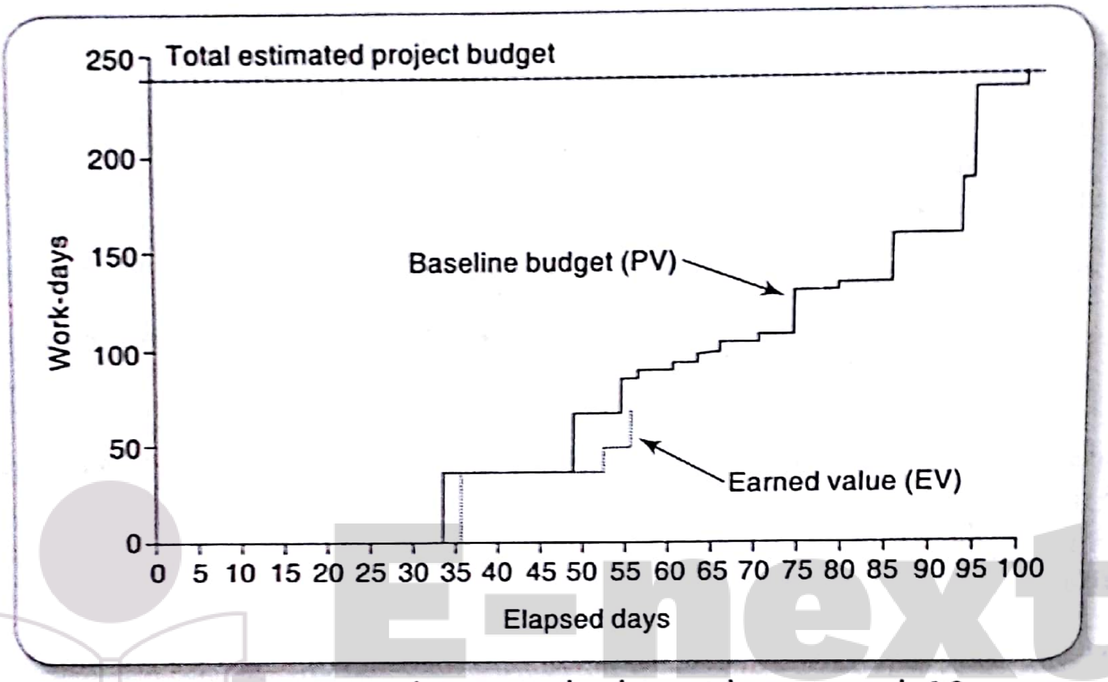

FIGURE 9.12 Amanda's earned value analysis at week 12

By studying Figure 9.13, can you tell exactly what has gone wrong with her project and what the consequences might be?

As well as recording EV, the actual cost of each task can be collected as *actual cost* (AC). This is also known as the *actual cost of work performed* (ACWP). This is shown in Figure 9.12, which, in this case, records the values as percentages of the total budgeted cost.

Figure 9.13 also illustrates the following performance statistics, which can be shown directly or derived from the earned value chart.

Schedule variance (SV)

The schedule variance is measured in cost terms as EV – PV and indicates the degree to which the value of completed work differs from that planned. Say, for example, that work with a PV of £40,000 should have been completed by now. In fact, some of that work has not been done so that the EV is only £35,000. The SV would therefore be £35,000 – £40,000, that is – £5,000. A negative SV means the project is behind schedule.

Time variance (TV)

Figure 9.13 also indicates the *time variance* (TV). This is the difference between the time when the achievement of the current earned value was planned to occur and the time now. In this case, the current EV

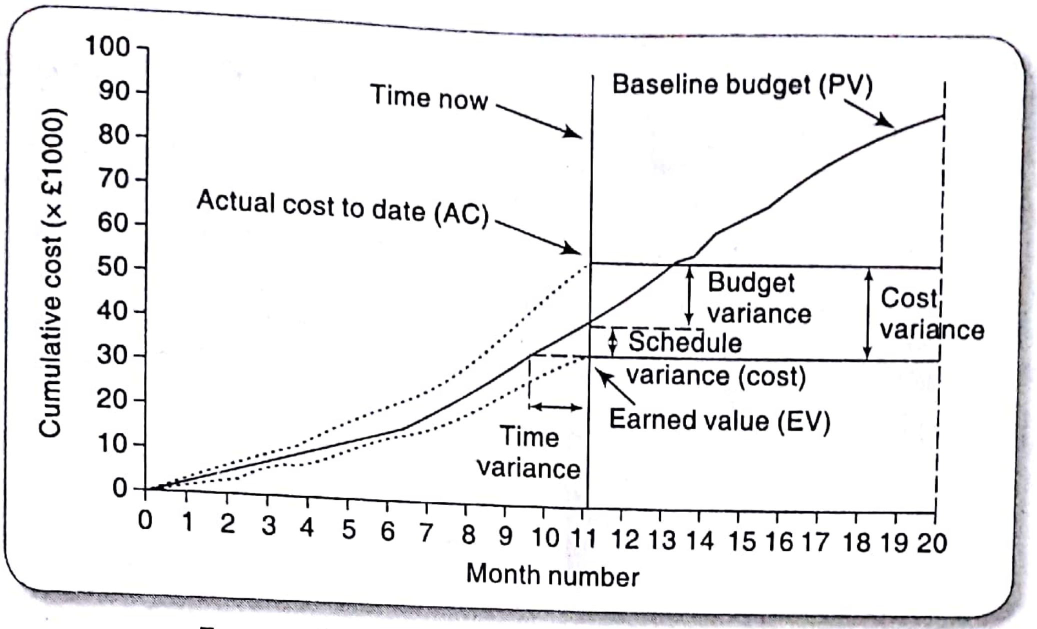

FIGURE 9.13 An earned value tracking chart

should have been achieved in the early part of month 9 and as the time now is the end of month 11, the TV is about –1.75 months.

Cost variance (CV)

This is calculated as EV – AC and indicates the difference between the earned value or budgeted cost and the actual cost of completed work. Say that when the SV above was calculated as –£5,000, £55,000 had actually been spent to get the EV. The CV in this case would have been £35,000 – £55,000 or –£20,000. It can also be an indicator of the accuracy of the original cost estimates. A negative CV means that the project is over cost.

Performance ratios

Two ratios are commonly tracked: the cost performance index (CPI = EV/AC) and the schedule performance index (SPI = EV/PV). Using the examples above, CPI would be £35,000/£55,000, that is, 0.64, and SPI would be £35,000/£40,000, that is, 0.88. The two ratios can be thought of as a 'value-for-money' indices. A value greater than one indicates that work is being completed better than planned, whereas a value of less than one means that work is costing more than and/or proceeding more slowly than planned.

CPI can be used to produce a revised cost estimate for the project (or *estimate at completion* – EAC). EAC is calculated as BAC/CPI where BAC (budget at completion) is the current projected budget for the project. If the BAC was £100,000 then a revised estimate at completion (EAC) would be £100,000/0.64 or £156,250. Similarly, the current SPI can be used to project the possible duration of the project given the current rate of progress. Say the planned total duration for the project is 23 months – in earned value terminology this is the *schedule at completion* (SAC). A time estimate at completion (TEAC) can be calculated as SAC/SPI. In this case it would be 23/0.88, that is, 26.14 months. This is only an approximate guide: where there are several parallel chains of activities being carried out concurrently – as we saw in Chapter 6 – the project duration will depend on the degree to which the activities that have been delayed are on the critical path.

In the same way that the expenditure analysis in Figure 9.9 was augmented to show revised expenditure forecasts, we can augment the simple earned value tracking chart with forecasts as illustrated in Figure 9.14.

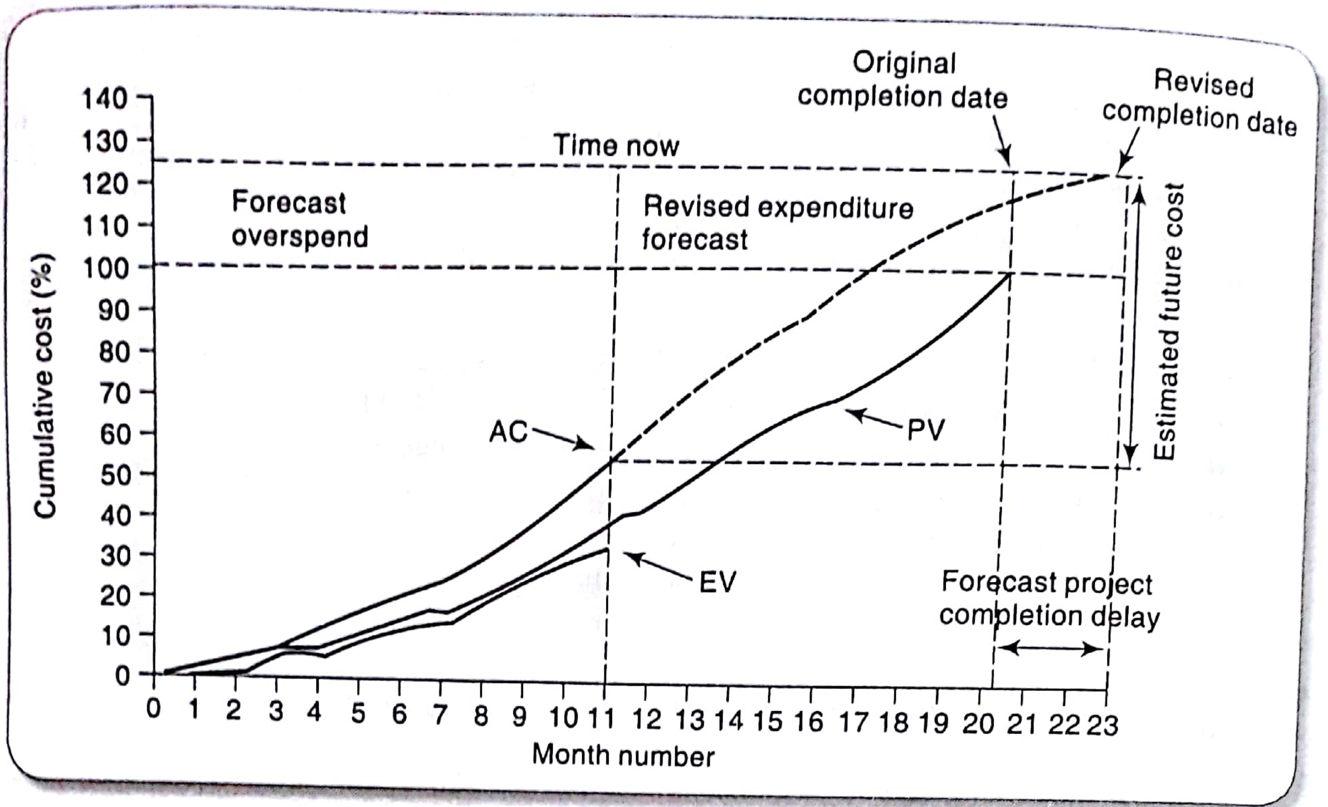

FIGURE 9.14 An earned value chart with revised forecasts

Earned value analysis has not yet gained universal acceptance for use with software development projects, perhaps largely because of the attitude that, whereas a half-built house has a value reflected by the labour and materials that have been used, a half-completed software project has virtually no value at all. This is to misunderstand the purpose of earned value analysis, which, as we have seen, is a method for tracking what has been achieved on a project – measured in terms of the budgeted costs of completed tasks or products.

Exercise 9.5

Suppose a project is to be completed in one year at the cost of £100,000. After three months, you realize that the project is 30% complete at a cost of £40,000. Assess the performance of the project.

9.8 Prioritizing Monitoring

So far we have assumed that all aspects of a project will receive equal treatment in terms of the degree of monitoring applied. We must not forget, however, that monitoring takes time and uses resources that might sometimes be put to better use!

In this section we list the priorities we might apply in deciding levels of monitoring.

- *Critical path activities* Any delay in an activity on the critical path will cause a delay in the completion date for the project. Critical path activities are therefore likely to have a very high priority for close monitoring.

- *Activities with no free float* A delay in any activity with no free float will delay at least some subsequent activities even though, if the delay is less than the total float, it might not delay the project completion date. These subsequent delays can have serious effects on our resource schedule as a delay in a subsequent activity could mean that the resources for that activity will become unavailable before that activity is completed because they are committed elsewhere.

Free float is the amount of time an activity may be delayed without affecting any subsequent activity.

- *Activities with less than a specified float* If any activity has very little float it might use up this float before the regular activity monitoring brings the problem to the project manager's attention. It is common practice to monitor closely those activities with less than, say, one week free float.
- *High-risk activities* A set of high-risk activities should have been identified as part of the initial risk profiling exercise. If we are using the PERT three-estimate approach we will designate as high risk those activities that have a high estimated duration variance. These activities will be given close attention because they are most likely to overrun or overspend.

PERT and the significance of activity duration variance was described in Chapter 7.

- *Activities using critical resources* Activities can be critical because they are very expensive (as in the case of specialized contract programmers). Staff or other resources might be available only for a limited period, especially if they are controlled outside the project team. In any event, an activity that demands a critical resource requires a high level of monitoring.

9.9 Getting the Project Back to Target

Almost any project will, at one time or another, be subject to delays and unexpected events. One of the tasks of the project manager is to recognize when this is happening (or, if possible, about to happen) and, with the minimum delay and disruption to the project team, attempt to mitigate the effects of the problem. In most cases, the project manager, at least initially, tries to ensure that the scheduled project end date remains unaffected. This can be done by shortening remaining activity durations or shortening the overall duration of the remaining project in the ways described in the next section.

A contingency plan should, of course, already exist as a result of the risk analysis methods described in Chapter 7.

It should be remembered, however, that this might not always be the most appropriate response to disruptions to a plan. There is little point in spending considerable sums in overtime payments in order to speed up a project if the customer is not overly concerned with the delivery date and there is no other valuable work for the team members once this project is completed.

The schedule is not sacrosanct – it is a plan that should be adhered to so long as it is relevant and cost-effective.

There are two main strategies to consider when drawing up plans to bring a project back on target – shortening the critical path or altering the activity precedence requirements.

Shorten the critical path

The overall duration of a project is determined by the current critical path, so speeding up non-critical path activities will not bring forward a project completion date. However, there are several ways in which this might be done.

- *Adding resources – especially staff* Exhorting staff to 'work harder' might have some effect, although frequently a more positive form of action is required, such as increasing the resources available for some critical activity. Fact-finding, for example, might be speeded up by allocating an additional

Time/cost trade-off: there is a general rule that timescales can be shortened by buying more (or more expensive) resources; sometimes this is true.

analyst to interviewing users. It is unlikely, however, that the coding of a small module would be shortened by allocating an additional programmer – indeed, it might be counterproductive because of the additional time needed for organizing and allocating tasks and communicating. While adding more staff may be able to speed up progress, this would be at an additional cost. In EV terms, negative schedule variance (SV) may be reduced, but at the price of increasing a negative cost variance (CV).

- *Increase use of current resources* Resource levels can be increased by making them available for longer. Thus, staff might be asked to work overtime for the duration of an activity and computing resources might be made available at times (such as evenings and weekends) when they might otherwise be inaccessible.
- *Reallocate staff to critical activities* The project manager might consider allocating more efficient staff to activities on the critical path or swapping resources between critical and non-critical activities. When a project is actually executed, the critical path may change as the actual durations of activities will vary from the original estimates and staff allocations may be adjusted to reflect this.
- *Reduce scope* The amount of work to be done could be reduced by reducing the scope of the functionality to be delivered. The client may prefer to have a subset of the promised features on time – especially if they are the most useful ones – rather than have the delivery of the whole application delayed.
- *Reduce quality* Some quality-related activities such as system testing could be curtailed. This would probably lead to more corrective work having to be done to the 'live' system once it has been implemented.

By such means we can attempt to shorten the timescale for critical activities until such time as either we have brought the project back to schedule or further efforts prove unproductive or not cost-effective. Remember, however, that shortening a critical path often causes some other path, or paths, to become critical (see Section 6.14).

Reconsider the precedence requirements

If attempting to shorten critical activities proves insufficient, the next step is to consider the constraints by which some activities have to be deferred pending completion of others. The original project network would most probably have been drawn up assuming 'ideal' conditions and 'normal' working practices. It might be that, to avoid the project delivering late, it is now worth questioning whether as yet unstarted activities really do have to await the completion of others. It might, in a particular organization, be 'normal' to complete system testing before commencing user training. In order to avoid late completion of a project it might, however, be considered acceptable to alter 'normal' practice and start training earlier.

One way to overcome precedence constraints is to subdivide an activity into a component that can start immediately and one that is still constrained as before. For example, a user handbook can be drawn up in a draft form from the system specification and then be revised later to take account of subsequent changes.

If we do decide to alter the precedence requirements in such a way, it is clearly important to be aware that quality might be compromised and to make a considered decision to compromise quality where needed. It is equally important to assess the degree to which changes in work practices increase risk. It is possible, for example, to start coding a module before its design has been completed. It would normally, however, be considered foolhardy to do so since, as well as compromising quality, it would increase the risk of having to redo some of the coding once the final design had been completed and thus delay the project even further.

Maintaining the business case

In making decisions about the management of the project, the main concern of the project sponsor, that is, the stakeholder who is putting up the money for the project, is whether the business case for the project has been preserved. You may recall from Chapter 2 that the value of the benefits of a project must be greater than its cost for the project to be viable. If costs increase, then this reduces the value of the benefits at the end of the project. If the project is delayed or the amount of functionality in the deliverables is curtailed, it means that the benefits that the project will generate will be reduced. At some point the costs may exceed the benefits and the project then loses its viability. A decision could then be made to cancel the project.

Exception planning

The project manager will normally be allowed to change the detail of a plan as long as the agreed project outcomes are produced on time and within budget.

In some cases, an operational change may affect other stakeholders. One such case would be where the timing of acceptance testing by users had to be changed. In such a case the project manager would need to gain the acceptance of these stakeholders for the change.

Some changes to the plan might have an impact on the delivery date, project scope or costs. These, in turn, could affect the business case for the project. Here the project manager would need to gain the approval of the business sponsors of the project. We saw above that the interests of the sponsors could be represented through a group variously known as a Project Board (in PRINCE2), project management board or steering committee.

One approach, adopted by PRINCE2, is to require the project manager to write an *exception report* that explains the reasons for the deviation from the existing plan. The consequences of the deviation should be detailed, and if possible a number of options for dealing with the problem. The probable impact of each option on the business case is projected, and a recommendation on a course of action is presented. The Project Board, or equivalent, having considered the report and having approved one of the options, may then task the project manager with producing a more detailed *exception plan*. If this is then approved it replaces the existing plan.

9.10 Change Control

So far in this chapter, we have assumed that the nature of the deliverables has not changed. A project leader like Amanda or Brigette might find, however, that requirements are modified because of changing circumstances or because the users get a clearer idea of what is really needed. The payroll system that Brigette is implementing might, for instance, need to be adjusted if the staffing structure at the college is reorganized.

Other, internal, changes will crop up. Amanda might find that there are inconsistencies in the program specifications that become apparent only when the programs are coded, and these would result in amendments to those specifications.

When a document such as the user requirements is being developed there may be many different versions of the document as it undergoes cycles of development and review. Any change control process at this point would be very informal and flexible. At some point what is assumed to be the final version will be created. This is *baselined*, effectively frozen. Baselined products are the foundation for the development of further products – for instance interface design documents may be developed from baselined user requirements. Thus any changes to the baselined document could have knock-on effects on other parts of the project. The Product

Flow Diagrams (explained in Chapter 3) indicate relationships between the products of a project where this is the case. For this reason subsequent changes to baselined documents need to be stringently controlled.

Exercise 9.6

A change in a program specification will normally be carried through into changes to the program design and then changed code. What other products might need to be modified?

Change control procedures

A simple change control procedure for operational systems might have the following steps:

1. One or more users might perceive a need for a modification to a system and ask for a change request to be passed to the development staff.
2. The user management would consider the change request and, if they approve it, pass it to the development management. It is important that there is a single authorized channel for *requests for change (RFCs)* between the client community and the management of the developers. There would be some filtering within the client community to ensure that the proposed change does genuinely provide a benefit before the RFC is generated.
3. There would be one person within the development area who would receive and process RFCs. They would delegate a member of staff to look at the request and to report on the practicality and cost of carrying out the change. The developer would, as part of this, assess the products that would be affected by the change.
4. The development representative would report back to the user management on the findings and the user management would decide whether, in view of the cost quoted, they wish to go ahead.
5. There would be some individual or group who represented the major stakeholders, both users and developers and also the project sponsor, who would have the authority to prioritize the RFCs for action. Ultimately this should be the Project Board or equivalent. However, the large proportion of RFCs would be relatively small in scope. This could cause a bureaucratic bottleneck if all these changes had to be considered at the highest level. A smaller group of active stakeholder representatives might therefore be delegated the responsibility for considering and approving changes up to a certain level of expenditure. This group would adopt the role of a *change control board*, although they might not actually be called that. A further step is to give the project managers allowances that would allow them accept minor changes (as long as they are documented with an RFC, etc.) as long as they do not exceed planned cost and delivery targets. There is thus a general principle that the larger the amendment the higher in the control hierarchy it would have to be reported. However, this is not simply a matter of size. A very large number of seemingly small changes could have a serious accumulative effect on project progress which may call for the attention of higher management. A very large set of changes might trigger the project manager to produce an exception report – see above.
6. Once an RFC has been approved for action, one or more developers are authorized to take copies of the master products that are to be modified. This would need to be done through the configuration librarian.
7. The copies are modified. In the case of software components this would involve modifying the code and recompiling and testing it.

8. When the development of new versions of the product has been completed the user management will be notified and copies of the software will be released for user acceptance testing.
9. When the user is satisfied that the products are adequate they will authorize their operational release. The master copies of configuration items will be replaced.

Exercise 9.7

The above steps relate to changes to operational systems. How could they be modified to deal with systems under development?

Changes in scope of a system

A common occurrence with IS development projects is for the size of the system gradually to increase. One cause of this is changes to requirements that are requested by users.

This is sometimes called scope creep.

Exercise 9.8

Think of other reasons why there is a tendency for scope creep.

The scope of a project needs to be carefully monitored and controlled. One way is to re-estimate the system size in terms of SLOC or function points at key milestones.

Configuration librarian's role

Control of changes and documentation ought to be the responsibility of someone who may variously be named the configuration librarian, the configuration manager or the project librarian. Among this person's duties would be:

- The identification of all items that are subject to change control
- The establishment and maintenance of a central repository of the master copies of all project documentation and software products
- The setting up and running of a formal set of procedures to deal with changes
- The maintenance of records of who has access to which library items and the status of each library item (e.g. whether under development, under test or released)

BS EN ISO 9001:1994 (formerly BS 5750) requires that a formal change control procedure be in place.

It will be recalled that it was suggested that the setting up of change control procedures might be one of the first things the Brigette would want to do at Brightmouth College.

9.11 Software Configuration Management (SCM)

In the previous section, we examined some of the reasons why changes might occur in Amanda's project. Though the discussion was restricted to changes due to requirements change, the changes can take place in any of the work products and may be due to many reasons such as bug fix, changes on account of work simplification, efficiency considerations, etc. We have discussed how change management can be done manually by a designated configuration librarian. However, the manual change management process gets overwhelmed when we consider changes taking place on all work products and when there are multiple variants of the product. In this situation, a systematic software configuration management (SCM) process with appropriate tool support needs to be deployed.

SCM is concerned with tracking and controlling changes to the software. In any systematic development and maintenance environment, various work products (code, design document, code, etc.) associated with the software continually change during the development as well as the maintenance phase. In a team development environment, each member of the development or maintenance team would be assigned to handle some modification requests. Therefore every work product would have to be accessed and modified by several members. In such a situation, unless a proper configuration management system is deployed, several problems can appear. We first discuss the context in which these problems appear, and subsequently we shall investigate the different problems that a development team might face if it does not deploy an effective configuration management system. Finally, we discuss the configuration management process.

Context in which configuration management is necessary

During the development phase, the work products get modified as development activities are carried out. During the maintenance phase, the work products change due to various types of enhancements and adaptations that are carried out including bug fixes. Thus, the state of the work products continually change both during the development as well as maintenance phase. The state of all work products at any point of time is called the configuration of the software product. Software configuration management deals with effectively tracking and controlling the configuration of a software product during its entire life cycle. For effective configuration management, it is necessary to deploy a configuration management tool. Thus, we can say that the different concepts associated with configuration management are carried out in a project with the help of a tool. There are many configuration management tools available, some are open software that is free of any licensing fees and others are commercial tools. At the end of this section, we review a few open software configuration management tools.

Configuration management practices include version control and the establishment of baselines. Before we discuss configuration management, we must clearly understand terms like version, revision, variant, and baseline.

Few terminologies

In the following section we define terms like configuration, version, revision, variant, and baseline.

Configuration The configuration of software is the state of various work products that are under configuration control. The work products that are under configuration control are usually referred to as the configuration items. It is convenient to think of a configuration as a set of files representing various work products. For example, the configuration of a sample software product shown in Figure 9.15 consists of the configuration items (work products) $W1$, $W2$, … , Wn.

Version As development and maintenance activities are carried out on a software product, its configuration (that is, one or more configuration items) keeps changing. It often becomes necessary to refer to the configuration that existed at certain point of time. For example, we can say that refer to the last week's configuration of the software. Therefore, a version is a configuration that existed at certain point in time. More technically, versioning is a numbering scheme that helps us identify a specific configuration at a certain point in time. This is achieved by a configuration management tool by tagging the files resenting the configuration items with the version name.

Revision A revision system is a numbering scheme that is used to identify the state of a configuration item at any time. Each time a work product is updated its state changes. Thus, we can think of a work product going through a series of updates till it reaches a desired state. The successive states of a work product are its successive revisions. Thus each time a configuration item is updated, a new revision gets formed. It becomes possible to refer to a specific state of a work product by using its revision number.

Baseline A baseline is a software configuration that has been formally reviewed and agreed upon, and serves as a basis for further development.

Variant Variants are versions that are intended to coexist. Different variants may be needed to run the software on different operating systems or on different hardware platforms. For example, one variant of a mathematical computation package might run on Unix-based machines, another on Microsoft Windows machines. Variants may also be required to be created when the software is intended to be used with different levels of sophistication of the functionalities (e.g., novice version, enterprise version, professional version, etc.). Variants are often created during the operation phase during the development phase, and as and when software products with overlapping functionalities are required. Even the initial delivery of software might consist of several versions and more variants may be created later.

In the following, we first discuss the necessity of configuration management and subsequently we discuss the configuration management activities and tools.

Purpose of software configuration management

There are several reasons why proper configuration management of the work products in a project is essential. The following are some of the important problems that can occur if a proper configuration management system is not used.

- *Problems Associated with Concurrent Access* Possibly the most important reason for configuration management is to control the access to the different deliverable objects. Unless strict discipline is enforced regarding update and storage of different work products, several problems can appear. Let us assume that only a single copy of a program module is maintained, and several developers are working on it. Two developers may simultaneously carry out changes to the different functions of the same work product, and while saving overwrite each other.
- *Undoing Changes* It becomes easy to undo some part of a revision or even rollback development to a certain version. Unless proper configuration management system is in place, it becomes very difficult to undo a change.
- *System Accounting* System accounting denotes keeping track of who made a particular change to a configuration item, what change was exactly made, and when the change was made. Knowing the what, who, and when of changes will help in understanding why changes were made and whether some changes are redundant or for comparing the performance of particular versions. It may at times be required to rollback to a previous baseline if a change is not justified or is improper. Users may wish to compare today's version of some software with yesterday's version or last year's version. Since a configuration management system keeps track of every version and revision, this becomes a simple task.

- *Handling Variants* As we have already discussed, it often becomes necessary to create variants. In this situation, without a configuration management system, keeping track of all variants, their versions and revisions is a nontrivial task. Further, existence of variants of a software product causes some peculiar problems. Suppose you have several variants of the same module, and find that a bug exists in one of them. Then it has to be fixed in all versions and revisions. To do it efficiently, you should not have to fix it in each and every version and revision of the software separately. Making a change to one program should be reflected in all relevant versions and revisions.
- *Accurate Determination of Project Status* Normally, a project manager performs the configuration management activity by using a configuration management tool. In addition, a configuration management tool helps to keep track of various deliverable objects so that the project manager can quickly and unambiguously determine the current state of the project. The configuration management tool enables the developer to change the various components in a controlled manner.
- *Preventing Unauthorized Access to the Work Products* Configuration management helps implement a controlled change process. It therefore becomes possible to prevent unauthorized changes to the work products.

Configuration management process

Configuration management is carried out through the following two principal activities:

- *Configuration Identification:* This activity involves deciding which parts of the system should be kept under configuration management.
- *Configuration Control:* This activity is used to ensure that changes to a system occur smoothly.

In the following section, we provide an overview of these two activities.

- *Configuration Identification*

  Project managers normally classify the work products associated with a software development process into three main categories, viz., controlled, pre-controlled, and uncontrolled. Controlled work products are those that are put under configuration control. The team members must follow some formal procedures to change these. Pre-controlled work products are not yet under configuration control, but will eventually be under configuration control. Uncontrolled work products will not be subject to configuration control. Controllable work products include both controlled and pre-controlled work products.

Typical controllable work products include the following:

- Requirements specification document
- Design documents
- Tools used to build the system such as compilers, linkers, lexical analysers, parsers, etc.
- Source code for each module
- Test cases
- Problem reports

Exercise 9.9

What are the advantages and disadvantages of putting all the work products in a project under configuration control?

- *Configuration Control*

Configuration control is part of a configuration management system that most directly affects the day-to-day operations of developers. Configuration control allows only authorized changes to the controlled objects and prevents unauthorized changes. The project manager can give permission to some members to be able to change or access specific work products.

In order to change a controlled work product such as a code module, a developer can get a private copy of the module through a reserve operation (see Figure 9.15). Configuration management tools allow only one team member to reserve a module at any time. Once a work product is reserved, it does not allow anyone else to reserve this module until the reserved module is restored. Thus, by preventing more than one developer to simultaneously reserve a module, the problems associated with concurrent access are taken care of.

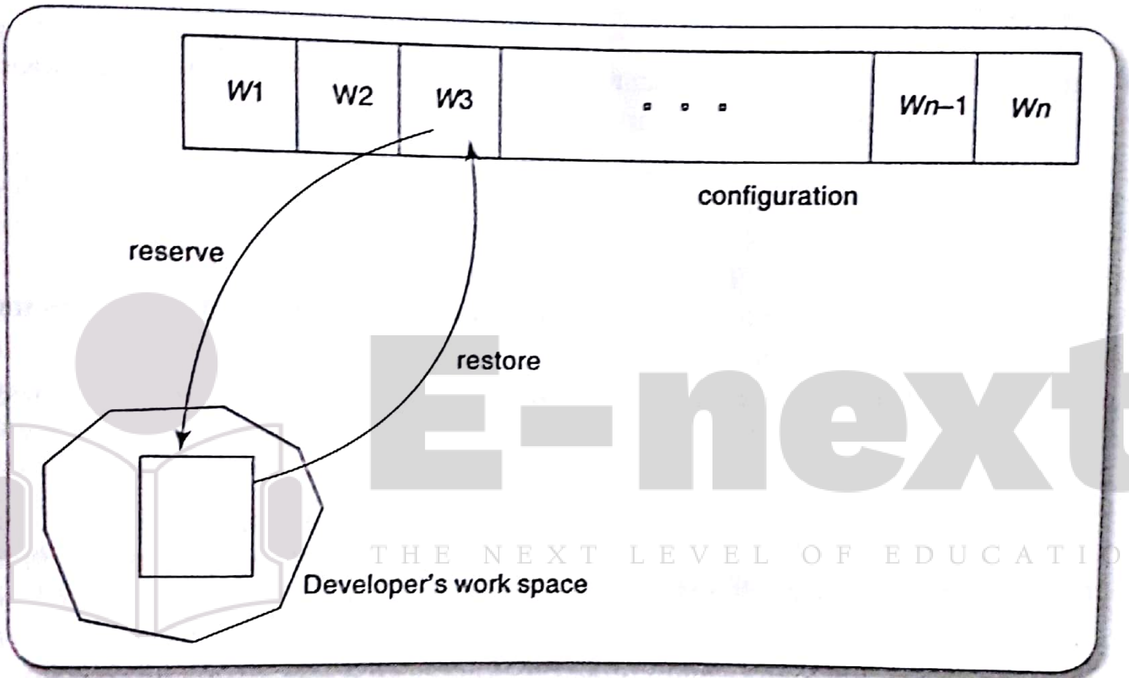

FIGURE 9.15 Work product modifications under configuration management

Modifications to a work product under configuration control

When developers need to change a work product they first make a reserve request. A reserve request by a team member is honoured only if appropriate authorization has been given by the project manager to that member for the specific work product. After the reserve command successfully executes, a private copy of the work product is created in their local directory. Then, they can carry out all necessary changes to the work product on their private copy. Once they have satisfactorily completed all necessary changes, the changes need to be restored in configuration management repository. However, restoring the changed work product to the system configuration requires the permission of a change control board (CCB).

The CCB is usually constituted from among the development team members. For every change that needs to be carried out, the CCB reviews the changes made to the controlled work product and certifies certain aspects about the change such as

- Change is well-motivated
- Developer has considered and documented the effects of the change

- Changes interact well with the changes made by other developers
- Appropriate people (CCB) have validated the change, e.g., someone has tested the changed code, and has verified that the change is consistent with the need

The change control board (CCB) is seldom a group of people. Except for very large projects, the functions of the change control board are normally discharged solely by the project manager or some senior member of the development team. Once the CCB reviews the changes to the module, the project manager updates the old configuration item through a restore operation (see Figure 9.15). A configuration control tool does not allow a developer to replace a work product in the configuration with his local copy unless he gets an authorization from the CCB. Therefore, incompletely modified or improperly modified work products cannot be updated in the configuration.

Release Management

It is desirable for a software development project to deploy a suitable release management process. A release management process systematizes the work carried out by the developers to provide a new release of a software and on the part of the users to smoothly and effortlessly obtain and use a new release. The release management process has become especially important after it has become possible for the users to instantly and effortlessly download new releases of a software over the Internet. The release process should involve minimal effort on the part of the developer to upload a new release of a software and on the part of the users to effortlessly download and install it. For example, when a new version of a system is to be released, a tool should automatically determine the changed components, the dependencies, and all interdependent components should be retrievable as a group, so that there is no possibility of inconsistency. There would be significant chance for inconsistency and a considerable amount of work to be carried out by the user, if the user would have to decide which specific components need to be downloaded. Further, retrievals of unnecessary and unchanged components should be avoided.

Open source configuration management tools

SCCS and RCS are two popular configuration management tools available on most UNIX systems. SCCS or RCS can be used for controlling and managing different versions of text files. SCCS and RCS do not handle binary files (i.e., executable files, documents, files containing diagrams, etc.). SCCS and RCS provide an efficient way of storing versions that minimize the amount of occupied disk space. Suppose, a module MOD is present in three versions MOD1.1, MOD1:2 and MOD1.3, then SCCS and RCS stores the original module MOD1.1 together with changes needed to transform MOD1.1 into MOD1.2, and MOD1.2 to MOD1.3. The changes needed to transform each baseline file to the next version are stored and are called deltas. The main reason behind storing the deltas rather than storing the full revision files is to save disk space.

The change control facilities provided by SCCS and RCS include the ability to incorporate restrictions on the set of individuals who can create new versions, and facilities for checking components in and out (i.e., reserve and restore operations). Individual developers check out components and modify them. After they have made all the necessary changes to a component, and after these changes have been reviewed, they check in the changed module into SCCS or RCS.

Conclusion

In this chapter we have discussed the requirements for the continual monitoring of projects and the need for making progress visible. Among the important points to emerge were:

- Planning is pointless unless the execution of the plan is monitored
- Activities that are too long need to be subdivided to make them more controllable
- Ideally, progress should be measured through the delivery of project products
- Progress needs to be shown in a visually striking way, such as through bar charts, in order to communicate information effectively
- Costs need to be monitored as well as elapsed time
- Delayed projects can often be brought back on track by shortening activity times on the critical path or by relaxing some of the precedence constraints

Further Exercises

1. Take a look at Amanda's project schedule shown in Figure 8.7. Identify those activities scheduled to last more than three weeks and describe how she might monitor progress on each of them on a fortnightly or weekly basis.
2. Amanda's Gantt chart at the end of week 17 (Figure 9.5) indicates that two activities are running late. What effect might this have on the rest of the project? How might Amanda mitigate the effects of this delay?
3. Table 9.2 illustrates Amanda's earned value calculations based on work-days. Revise the table using monetary values based on the cost figures that you used in Exercise 8.5. Think carefully about how to handle the costs of Amanda as project manager and the recovered overheads and justify your decisions about how you treat them.
4. If you have access to project planning software, investigate the extent to which it offers support for earned value analysis. If it does not do so directly, investigate ways in which it would help you to generate a baseline budget (PV) and track the earned value (EV).
5. Describe a set of change control procedures that would be appropriate for Brigette to implement at Brightmouth College.
6. Give examples of errors that can be identified in a design review.
7. Give examples of how project termination review results can change the development process and the project management process.
8. Suppose a project is budgeted to cost £150,000. The project is to be completed in 18 months. After two months, the project is 10% complete at an expense of £25,000. It was planned that after two months, 15% of the project work should have been completed. Compute the cost performance index and the schedule performance index. Interpret these values to assess the progress of the project.
9. What problems are you likely to face if you are developing several versions of the same software product according to a client's request and are not using any configuration management tools?

10. What do you understand by software configuration? What is meant by software configuration management? How can you manage software configuration (only mention the names of the principal activities involved)? Why is software configuration management crucial to the success of large software product development projects (write only the important reasons)?

11. What is a baseline in the context of software configuration management? How do baselines get updated to form new baselines?

12. How the following can be prevented while using a configuration management tool? Explain.
    (a) Two team members overwriting each other's work
    (b) Accidental deletion of work product
    (c) Unauthorized modifications to a work product

13. For each of the following questions, exactly one option is correct. Select the appropriate option.
    (i) Which one of the following most closely characterizes software configuration management?
        (a) It is used to ensure that all the artefacts associated with the project are correct and complete
        (b) It is used to break down the work parts into manageable chunks
        (c) It is used to ensure that all the project design criteria are met
        (d) It is used to ensure that all the artefacts associated with the project are consistent and up-to-date
    (ii) Which one of the following documents usually contains the change management?
        (a) Scope management plan
        (b) Communications management plan
        (c) Configuration management plan
        (d) Quality management plan
    (iii) At any time during a project execution, Earned Value (EV) helps to provide answer to which one of the following questions?
        (a) What is the value of work that should have been completed to date?
        (b) What is the value of work that has been completed to date?
        (c) How much money has been spent to date?
        (d) How much money should have been spent to date?
    (iv) Which one of the following most accurately describes configuration management?
        (a) Control of changes to project schedules
        (b) Formal review and acceptance of proposed changes to the project deliverables
        (c) Quality control of project deliverables
        (d) Controlling changes to the project deliverables
    (v) Which one of the following is an important purpose of software configuration management?
        (a) Support efficient customer relationship management such as providing technical help and accepting bug reports
        (b) Avoid maintaining multiple copies of the same project files
        (c) Keep track of which team member would perform which task and by when
        (d) Manage the workflow across the team members

(vi) Which one of the following would equal EV/PV?

(a) SPI
(b) CPI
(c) SV
(d) CV

(vii) Which one of the following scenarios can occur even while a software development organization is impeccably using a configuration management tool?

(a) Two team members overwrite each other's code that are under configuration control
(b) Accidental deletion of a work product that is under configuration control, by a developer and consequent loss of work
(c) A user reporting a software configuration error during installation
(d) Unauthorized modifications to a work product under configuration control by one of the developers

(viii) Which one of the following most closely describes configuration management in a software development project?

(a) Management of the configuration parameter settings in a developed software
(b) Management of objects that control the system configuration parameter settings
(c) Management of the states of various project deliverables
(d) Configuration of the management activities depending on the type of the project

(ix) Preparation of the project baseline budget brings out the financial implications for the project. Which one of the following is the project base line budget?

(a) Actual cost (AC)
(b) Earned value (EV)
(c) Planned value (PV)
(d) Cost performance index (CPI)

(x) The cost performance index (CPI) is given by which one of the following expressions?

(a) AC/EV
(b) $EV \times AC$
(c) EV/AC
(d) $AC \times EV$

(xi) During the execution of a project, the project manager determined the schedule performance index (SPI) of the project to be less than 1.0. What does it indicate?

(a) The project is over budget and there can be financial shortfall.
(b) More work is getting completed than what was planned.
(c) The project is running behind the monetary value of work that was planned to be accomplished.
(d) The project is progressing satisfactorily.

(xii) Suppose for a certain project budgeted for £2000, at certain time during the execution of the project, the project manager determined EV = £500 and AC = £400. What is the current estimated cost of the project?

(a) £2500
(b) £1800
(c) £1600
(d) £1200