Managing Small Projects

BUTLER COX
P.E.P

PEP Paper 18, May 1991

BUTLER COX
P.E.P

Managing Small Projects

PEP Paper 18, May 1991
by Chris Woodward

Chris Woodward

Chris Woodward is a principal consultant with Butler Cox in London, with 24 years of experience in the management of information systems. He is responsible for running the productivity assessment service provided as part of PEP. He was the author of PEP Paper 7, *Influence on Productivity of Staff Personality and Team Working*, PEP Paper 12, *Trends in Systems Development Among PEP Members*, and a report entitled *Managing the Human Aspects of Change*, published by the Butler Cox Foundation.

During his time with Butler Cox, he has also carried out a wide range of consulting assignments. Recent projects in which he has been involved include technology and strategy studies in the field of office systems, organisation studies, investigations leading to the selection of hardware and software, and work on the human aspects of systems development.

Prior to joining Butler Cox as one of the company's founders in 1977, Chris Woodward worked for four years with Citibank as a project manager with the European management services function. Earlier, with a subsidiary of Joseph Lucas, he was responsible for development projects. His early career was as a systems engineer with ICL.

He has a BSc honours degree in electrical engineering from Imperial College, London University.

BUTLER COX

Published by Butler Cox plc
Butler Cox House
12 Bloomsbury Square
London WC1A 2LL
England

Printed in Great Britain by Flexiprint Ltd., Lancing, Sussex.

BUTLER COX
P.E.P

Managing Small Projects

PEP Paper 18, May 1991
by Chris Woodward

Contents

BUTLER COX

Chapter 1

Small projects are reducing the overall level of performance

Small projects, both new developments and maintenance, make up a significant proportion of the work of many PEP members' systems development departments, and there are clear indications that this proportion is increasing, and is likely to continue to do so. Our research reveals, however, that most small projects undertaken by PEP members (excluding package implementations) perform at distinctly lower levels of productivity than large projects.

Development managers are not aware of the problems caused by small projects

Many systems development departments are probably not aware either of the size or of the nature of the problem. From PEP assessments, many members understand how some of their projects are performing, yet few have any real understanding of their productivity across the size range, and particularly at the lower end. Small projects are not simply downsized large projects. They are, for example, typically constrained by dependence on other projects, subject to tighter deadlines and frequent scheduling changes, more susceptible to slippage, and less resilient to unexpected problems. They therefore need to be managed and organised differently if systems managers are to make the most effective use of limited development resources.

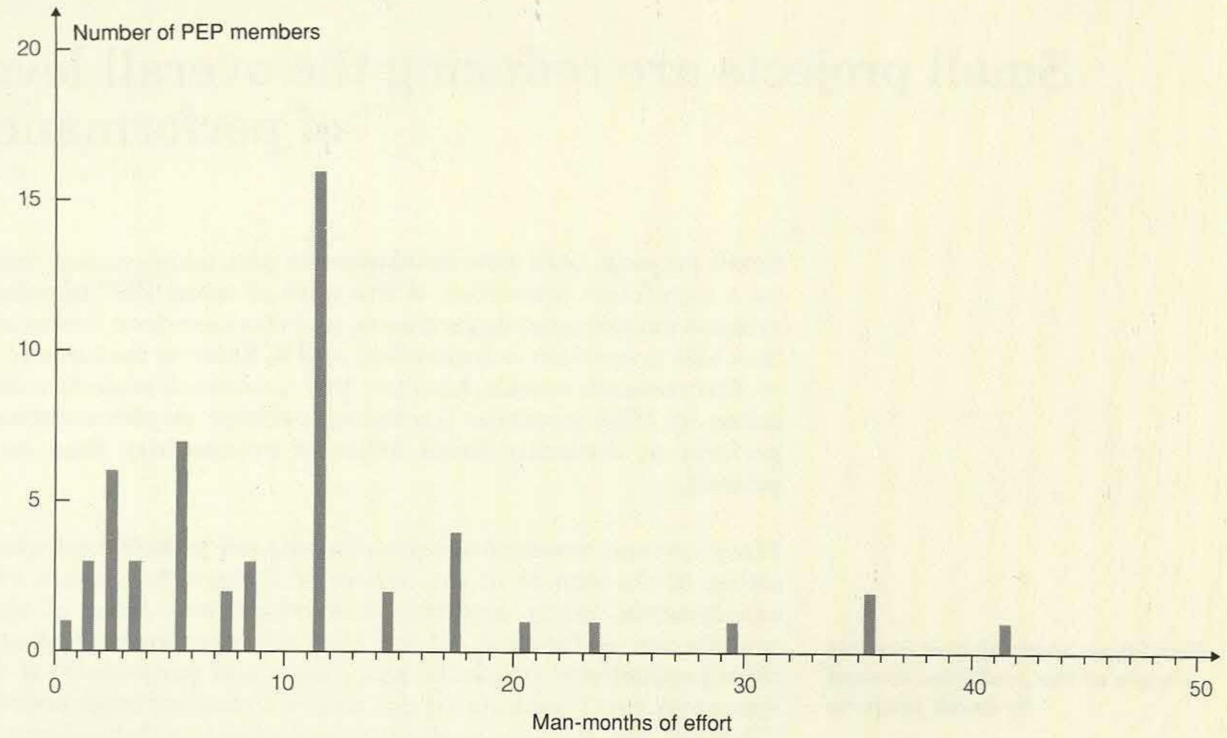
Small projects constitute a growing proportion of systems development work

PEP members generally define 'small' in terms of man-months of effort. Their definitions ranged from 1 to 42 man-months, with 12 man-months being the most common, as Figure 1.1, overleaf, illustrates.

A small project is up to 20,000 lines of code (or 350 function points) and requires less than 24 man-months of effort

For the purposes of this paper, we define 'small' to be any project up to 20,000 lines of code (about 350 function points for the average mix of languages typical of PEP projects), requiring not more than 24 man-months of effort, and usually undertaken by a team of not more than three people. Some projects of less than 20,000 lines of code may, of course, require more than 24 man-months of effort and larger teams, and such projects would certainly have low productivity. However, the combination of measures that we have chosen to define small projects reflects the average effort used to deliver an application of 20,000 lines of code. We include both projects to develop new systems, and maintenance projects of all types — corrective, adaptive and perfective. (These terms were defined in PEP Paper 8, *Managing Software Maintenance*.)

Figure 1.1 PEP members' definitions of a small project vary widely



(Source: Survey of PEP members)

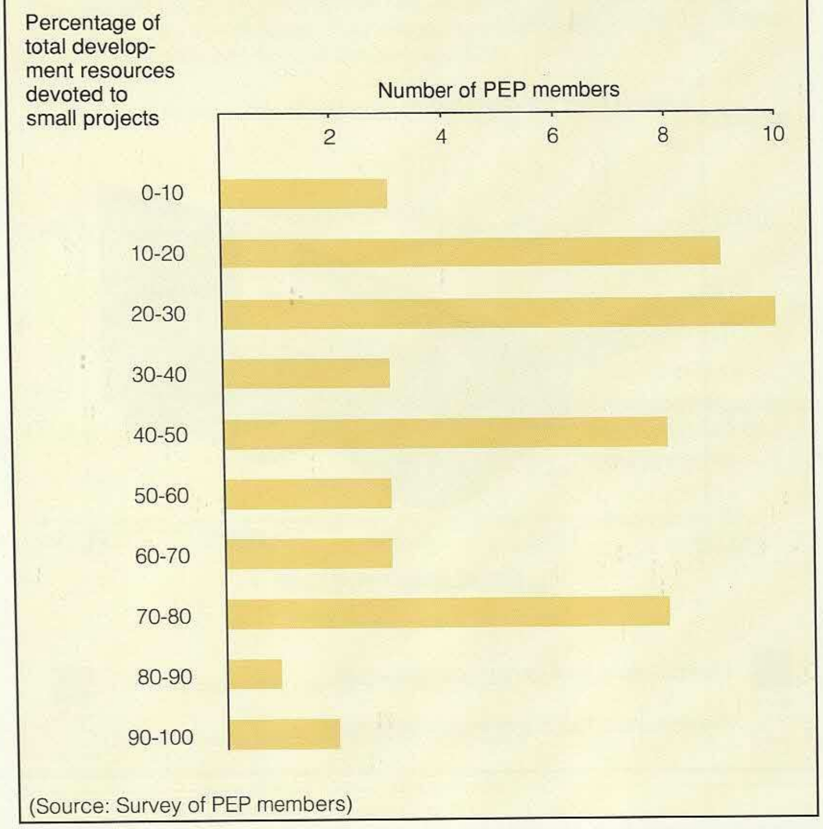
Small projects occupy 40 per cent of development staff time

In overall terms, small projects occupy about 40 per cent of the time of PEP members' development staff. (This corresponds with the observation that, at the time of writing, about 40 per cent of the 900 projects in the PEP database are small by our definition.) However, the proportion of development staff engaged in small projects varies widely, from less than 10 per cent in three organisations to 100 per cent in two organisations (see Figure 1.2). The number of development staff in the 50 PEP members we surveyed ranged from 25 to 800, with an average of about 150. Larger development departments have more staff working on small projects in absolute terms, but fewer as a proportion of total development staff — there is a tendency for the proportion of staff involved in small projects to decrease as departmental size increases.

The proportion of small projects is increasing

Since the start of PEP, we have observed a decrease in the size of PEP projects, expressed in lines of code. We reported in PEP Paper 12, *Trends in Systems Development Among PEP Members*, that the average size of project had decreased from about 45,000 to 35,000 lines of code between 1986 and 1987. Up to the beginning of 1991, the typical project size has continued to decrease, to about 31,000 lines of code. This reduction in size is accompanied, and at

Figure 1.2 The proportion of development staff engaged in small projects varies widely



(Source: Survey of PEP members)

least partly explained, by the use of fourth-generation languages, which has increased by 50 per cent since 1988, and now accounts for about 15 per cent of all code represented by the projects in the PEP database. We believe that the downward trend in the size of projects will continue into the foreseeable future.

Many small projects perform less well than other projects

There is enormous variation in the performance of small projects, expressed in terms of delivery rate per man-month, lines of code or function points. On average, however, small PEP projects perform less well than larger projects. There are, of course, some exceptions. For example, one PEP member's use of Pagefit on small PC developments is leading to rates of delivery as high as 50 function points per man-month.

The range of performance 'fans out' as project size increases

Our analysis of the PEP database shows that delivery rates rise steadily as system size increases from a low level up to a point where the range of performance 'fans out'. Thereafter, the performance of some projects continues to increase as size increases, whereas others perform less well. Figures 1.3 and 1.4, on pages 4 and 5, show the range of delivery rates (lines of code per man-month and function points per man-month respectively) for projects of a given size. Two sets of data are shown on each

Figure 1.3 The range of delivery rate for lines of code increases as project size increases

The diagram shows the minimum and maximum delivery rates for projects recorded in the PEP database.

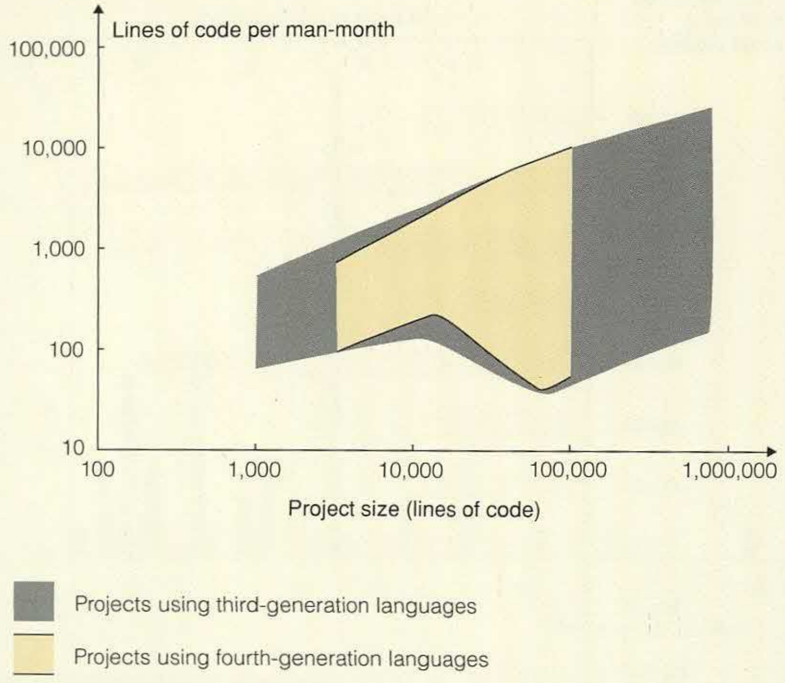


diagram: new development and maintenance projects in third-generation languages, and new development and maintenance projects in fourth-generation languages.

In terms of lines of code per man-month, the point at which performance changes from a generally increasing rate with increasing size ranges from 10,000 to 20,000 lines of code. Maintenance projects in third-generation languages, however, appear to reach the fan-out point earlier.

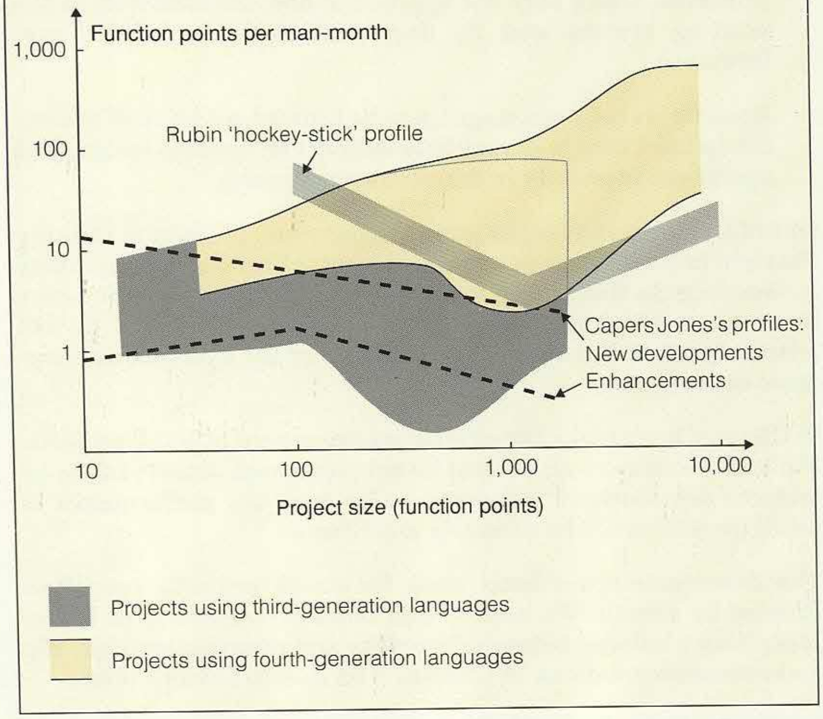
In terms of function points per man-month, the profile is similar, but projects using fourth-generation languages show a smaller range of delivery rate beyond the fan-out point. For development and maintenance projects in third-generation languages, the fan-out point appears to be about 100 function points, and for projects using fourth-generation languages, it appears to be between 500 and 800 function points.

Figure 1.4 also shows profiles published by Rubin Systems Inc (the 'hockey-stick' profile) and by Capers Jones of Software Productivity Research Inc (SPRI), for both new developments and maintenance. The SPRI profile for maintenance follows quite closely the lower edges of the PEP profile for traditional languages, while the profile for new developments conflicts with the PEP data for small projects (10 to 20 function points). The Rubin 'hockey-stick' profile suggests falling performance up to 1,000 function points in size and rising performance thereafter. If the Rubin and SPRI profiles are accurate reflections of performance for small

There is evidence that US performance on small developments is better than PEP members'

Figure 1.4 The range of delivery rate for function points increases as project size increases

The diagram shows the minimum and maximum delivery rates for projects recorded in the PEP database. Performance profiles identified by other researchers are shown for comparative purposes.



new developments, some organisations in the United States are achieving higher levels of performance than PEP members.

Few systems managers understand the unique requirements of small projects

The problems associated with the day-to-day management of small projects are often very different in emphasis from those associated with larger projects. For example:

— Dependence on other projects is often greater and can therefore have a greater influence on the progress and performance of small projects.

— Deadlines may often be tighter. Small projects may bring relatively large and visible benefits, and pressure can be high to implement them quickly.

— Project-management overheads can be quite significant for small projects. Unnecessary levels of bureaucracy and formality can constrain performance.

— Formal meetings may not be necessary on two- or three-person teams, where members are constantly 'rubbing shoulders' with each other.

- Scheduling plays a more prominent role, especially with small maintenance projects, and flexibility in scheduling is often critical to the effective use of staff. Slippage is often more prevalent.
- Small projects are usually less resilient to unexpected problems. Users may not appreciate how the condition of the existing system and its documentation can inhibit performance.
- Boundaries between stages may be blurred, and if staff are not disciplined, they may rush into coding, and systems testing and operation, especially in maintenance projects.

One of the main challenges in managing small projects is striking the right balance between over-burdensome bureaucracy and risky *laissez-faire*. In this report, we provide guidance for PEP members on managing and organising small projects so that they do not adversely affect the overall performance of the systems development department.

In Chapter 2, we make the case for measurement of small projects. Without measurement, even at its simplest level, actions taken by systems development managers to improve the performance of small projects could be seriously misdirected.

The development methods used for small projects are often adopted by default. We believe that more care needs to be taken in striking a balance between formality and unnecessary risk. We make recommendations in Chapter 3 on how to achieve this.

In Chapter 4, we deal with the organisational and human aspects of managing small projects. Good performance is likely to be achieved only when small projects are dealt with by specialist units and by managers who understand and manage the unique characteristics of small projects.

Research sources

The prime source of the information on which this paper is based is the data provided by PEP members. We circulated a questionnaire to all PEP members, and received 60 completed questionnaires from 50 organisations. We subsequently met representatives of 15 members and telephoned another 10, to discuss particular issues in more detail.

At the time of writing, the PEP database has data on about 900 projects, of which about 40 per cent are small, by our definition. We analysed this data to identify profiles of performance. We also obtained a sample of data for about another 100 small projects from eight PEP members. (These projects would not usually be included in the PEP database because of the criteria used to select projects for PEP assessment.) This data confirmed the pattern of performance shown by the small projects already held in the PEP database.

We also met three of the principal suppliers of development methods to discuss the particular problems that arise with small projects, and their existing and planned products. We talked to other organisations that have collected project data in order to ascertain whether their data on the performance of small projects confirmed our analysis of small PEP projects.

Chapter 2

Measure the performance of small projects

The lack of formality in undertaking small projects often hinders the ability of systems development managers to gather useful data. The problem is further complicated by the fact that performance can vary widely depending on the individual members of staff involved, and in the case of maintenance projects, on the particular program being altered.

PEP members need to establish and apply clear standards for measuring small projects, and to analyse the measures according to the individuals involved and/or the program being maintained. Regular analysis of the measurement data should help managers to make sound decisions about how to improve the performance of small projects. It will also help to identify the size of project at which peak performance is obtained.

Establish and apply clear standards for measuring small projects

Standards are needed pertaining to what is measured, when it is measured, and how it is measured. The basic measurement programme should collect aggregate data about effort (in man-days) and time (in weeks), split by the main development stages for new developments. As a minimum, these should be the three PEP stages of feasibility study, functional design and main build. For maintenance projects, the split should be, as a minimum, between analysis (sometimes referred to as problem analysis) and development. A measure of size should also be collected — either function points or lines of code, or both. The measurement programme should be comprehensive, which means collecting data for *all* projects; otherwise, development managers will not have a complete picture of how development resources are being used.

Data should be collected for the feasibility, functional-design and main-build stages, at least

The measures and their purpose need to be explained to staff, emphasising the importance of objectivity and accuracy. Staff must understand that the aim of measurement is to acquire a basis on which to improve performance, not to berate individuals. The manager who misuses information in this way will destroy the trust that is so necessary for effective and accurate measurement.

Measurement data should not be used to berate staff

The effort required to collect the basic effort and time measures for a small project is likely to range from less than an hour to half a man-day. Effort will also be required for counting function points. A counting rate of 1,000 function points per man-day is typical, although the quality of the documentation and the availability of people with knowledge of the system can cause considerable variations. Small projects could range from 10 to about 500 function points in size, so the effort required to count function points will range from less than an hour to three or four hours.

Information about key project characteristics should also be collected

The basic numeric data about the effort, duration and size should be supplemented by statements of key project characteristics (such as the classification of the type of project, the main technologies and methods used, the levels of experience of staff, and other easy-to-obtain factors, such as those used in PEP assessments). Measurement of small maintenance projects could also be supplemented by a measure of the size of the applications portfolio being maintained — preferably expressed in function points rather than in lines of code. The level of effort used to maintain applications can then be related not only to the number of function points changed or added, but also to the total portfolio, and thus help in setting staffing levels.

The distribution of effort between different types of activity should be measured

To manage the development processes for small projects more effectively, managers will also need to understand where effort is being used — that is, on what types of activity. This requires splitting the effort into categories such as management, quality assurance and testing. We believe that most PEP members are in a position to collect such data with only marginal extra effort; the main difficulty is a lack of discipline by staff in recording their time appropriately for small projects. Managers should encourage them to record their time accurately by scrutinising the data that is collected.

The process of collecting the measurement data for small projects should be part of the management-control mechanisms such as those to do with planning and progressing, change control, configuration control and so on. Introducing other mechanisms in addition to the control mechanisms would be burdensome and would probably be resented by systems staff.

Record and analyse information pertaining to individual staff and programs

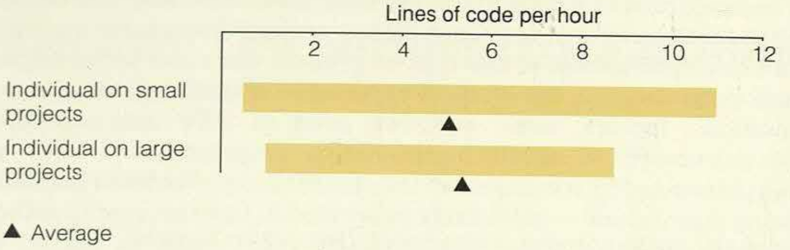
The value of measurement data depends to a large extent on its immediate analysis by responsible managers. Any anomalies should be dealt with as quickly as possible. If staff see that the information is recorded and forgotten, they are not likely to be rigorous about the way they collect it. When measuring small projects, managers need to pay particular attention to the contributions of individual members of staff, and in the case of maintenance projects, to the effort used to maintain individual modules or programs.

Record the contributions of individuals to help with estimating and planning small projects

Individual performance varies more widely on small projects

The performance of individuals is more varied on small projects than on large projects. This is demonstrated quite clearly by information published by NASA (see Figure 2.1, overleaf), which shows that the lines of code produced per hour can vary from 0.5 to 11 for small projects, compared with 1 to 8.75 for large projects. If measurements are to be used to support estimating for

Figure 2.1 The performance of individuals varies more widely on small projects than on large ones



(Source: NASA)

future small projects, managers need to keep track of the variability caused by individuals.

Many managers may consider that it is easier to estimate for small projects, as there are usually fewer unknowns, fewer changes in direction, and tighter definitions. However, unless a manager has measurements about small projects that include details of the variations attributable to individuals, he will not be in a position to estimate accurately for the future.

We recommend that individual contributions be recorded, in terms of effort by individual. Associations between individual contributions and project performance can then be maintained, analysed statistically and used to support estimating. The information obtained in the process should, of course, be treated confidentially, and not misused. The only purpose in passing it on to an individual is to help him to improve his own performance (or to make it more consistent), not to compare his performance with that of others.

Analyse maintenance effort by program

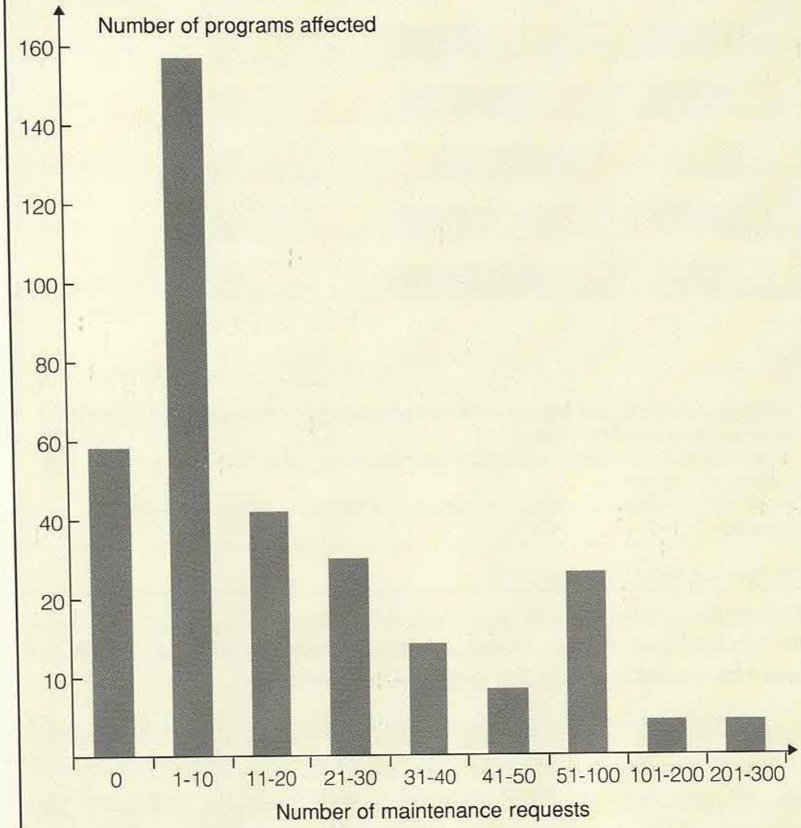
Maintenance activity is typically distributed unevenly over the programs within a system. Figure 2.2 shows maintenance data for a manufacturing system of nearly 350 programs. The programs varied in size from about 50 to 6,500 lines of code, with an average of about 1,200. Much of the variation in maintenance requests indicated in the figure could be explained by variations in program size, but a significant amount of the variation could not be explained in this way. It is likely that the maintainability of individual programs also varied.

To improve long-term productivity, it may be necessary to rewrite some existing programs

Knowing where maintenance activity is focused and the maintainability of individual programs, particularly those that are affected most often by changes, is crucial to the productivity of small maintenance projects. Improving productivity in the long term may necessitate rewriting programs that are maintained frequently and that are relatively difficult to maintain. Effective decision-making of this kind requires information to be recorded on the maintenance effort for program specification, coding and unit testing that is associated with each program being maintained, and on the size of the changes being made. About 40 per cent of PEP members we surveyed have time-recording systems that can identify the program or module being worked on. These organisations may be able to use this data for recording, and

Figure 2.2 Maintenance activity is typically distributed unevenly over the programs within a system

The diagram shows maintenance data for a manufacturing system of 346 programs.



(Source: Gremillion, L L. Determinants of program repair maintenance requirements. *Communications of the ACM*, vol. 27, no. 8, August 1984, p.826-832.)

subsequently analysing, the programs and effort that are associated with changes. Others will need to upgrade their time-recording systems.

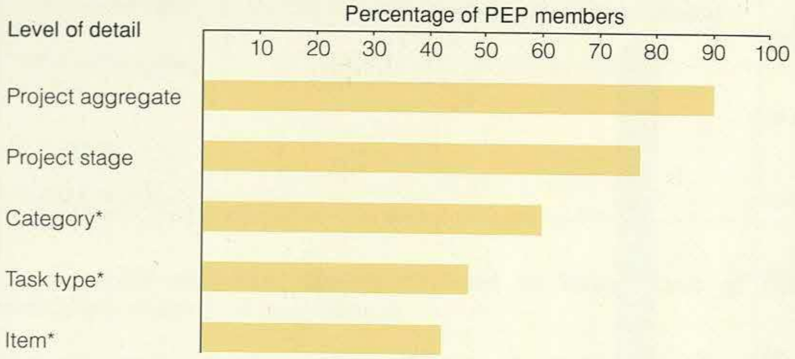
Analyse how effort is being used

Most PEP members analyse data from time-recording systems only at the aggregate project level

The level of detail of project information that PEP members collect in their time-recording systems is shown overleaf in Figure 2.3. Larger organisations tend to collect less detailed data, omitting information about task types (that is, activities such as entity modelling, data-flow diagramming, coding and unit testing) and items (that is, specific named deliverables of the work, such as the name of a module). Most PEP members appear to analyse this data only at the aggregated project level, however. This may be a reflection of the limited confidence that they have in their more detailed data. However, analysing the more detailed data could provide useful insights into where effort should be expended. For example, analysis of the relative effort used in systems testing and quality assurance may support the case for shifting more effort

Figure 2.3 Most PEP members collect aggregate project data but fewer than half collect detailed data about work items

Forty-one per cent of organisations collect data at all levels, although at the item level, it may be collected selectively. Three members collect data on man-months of effort only for those projects submitted to PEP for assessment.



* A 'category' is a broad type of activity such as project management, quality assurance or documentation.
A 'task type' is an activity such as entity modelling, data-flow diagramming, coding or unit testing.
An 'item' is a particular named deliverable of the work, such as the name of a module.

(Source: Survey of PEP members)

into quality assurance, resulting in a reduction in total effort for these two activities and for projects as a whole.

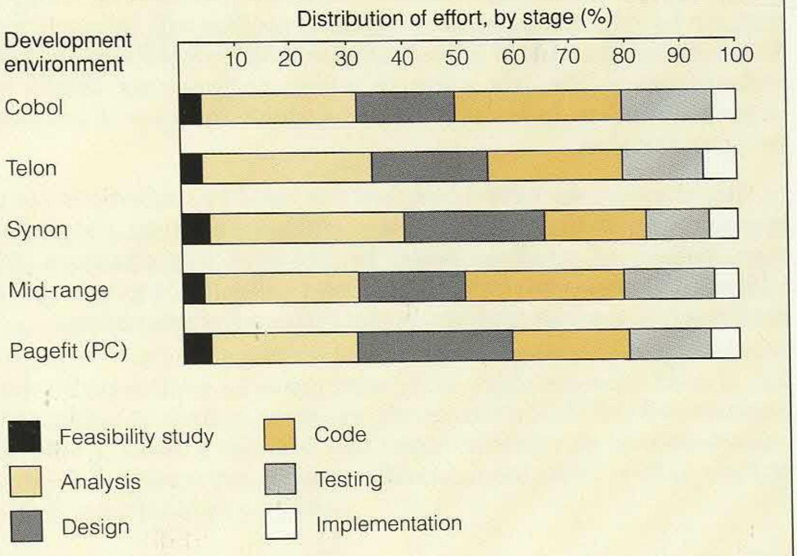
We believe that it is important for PEP members to exploit the information potential of their time-recording systems. To achieve this, 40 per cent of PEP members will need to enhance the capability of their systems to include, at least, category of work — that is, project management, quality assurance, documentation and so on. In particular, all PEP members should work to improve the accuracy of the data that they collect for small projects. They should periodically analyse the effort being used by small projects, particularly by category of work and by stage, as a basis for understanding how effort is currently spent and for effective decision-making about changing the way in which small projects are managed.

The accuracy of data collected for small projects should be improved

Analyse projects by language and machine

In analysing data on small projects, systems managers need to take account of the variations that arise in projects developed in different hardware and software environments. In PEP Paper 16, *Project Estimating*, we reproduced the analysis prepared by one member (Lloyd's of London, which provides services to members of the Lloyd's insurance market in London) that showed how the proportions of effort required at each development stage vary according to the programming language used. A similar analysis, produced by BOC (an industrial gases and health-care products and services group), is shown in Figure 2.4. Besides revealing details about the relationships between languages, machines and performance, such analyses are useful for supporting estimating. Individual projects that have widely different effort distributions

Figure 2.4 Analysis of how effort is distributed by stage of work will reveal relationships between performance and programming languages/machine size



(Source: BOC)

from the average pattern are the ones that might profitably be studied to learn from for the future.

Use statistical analysis to estimate problem-analysis effort in maintenance projects

It is difficult to estimate the problem-analysis effort of small maintenance projects

The proportion of effort required at the problem-analysis stage of small projects varies widely, but can be as high as 80 per cent. The effort required to implement the changes is, however, more predictable. The difficulty in estimating the problem-analysis effort arises from the fact that it is often not possible to formalise and partition maintenance work in the same way as new systems development work, which can be structured and broken down into sub-tasks.

The best way of estimating problem-analysis effort for maintenance projects is to carry out a statistical analysis on past project data, to identify the relationships between problem-analysis effort, size of change and development effort. The results of such an analysis can be used to predict the range of effort likely to be needed in both the analysis stage and the implementation stage.

Seek to identify the size of projects that maximise productivity

Although our interest in this paper is focused on small projects, it is also important that PEP members identify the relationship between project size and productivity in their particular environment. This can be achieved by collecting just the basic measures of effort, time and size for projects. In Figure 1.3 and Figure 1.4, we identified the general performance profiles for all PEP projects. PEP members need to identify and monitor the particular performance profiles of their own development environments. For example, PL/1 projects at Lloyd's of London usually range in size from 50 to about 1,000 function points. It has found that maximum productivity is obtained at about 500 function points. The insights that can be gained from examining these profiles will help systems development managers to decide how small projects should be undertaken, to identify areas in which performance might be improved, and to influence decisions about the size of projects being undertaken.

One organisation has found that maximum productivity is obtained on projects of about 500 function points

In this chapter, we have identified the need to implement some measures as a basis for effective decision-making about the management of small projects. In the next two chapters, we address the two main areas of opportunity for improving the performance of small projects — methods and organisation.

Chapter 3

Formalise methods for small projects

In most development departments, adopting and refining methods and tools to support large new systems developments attracts a great deal of management attention. In PEP Paper 8, we reported the results of a Butler Cox survey of PEP members, in which 70 per cent of managers rated new systems development as being more demanding of their time than maintenance. This is not surprising as such projects are often large and difficult to manage, involve greater risks and are more strategic in nature. Nevertheless, a greater proportion of development effort is often used on small maintenance projects.

Informal methods account for some of the poorer performance of small projects

The approaches and methods used for smaller projects are often less formal and structured than those used for large projects, and tend to be variable. In part, this may be attributed to an implicit understanding that too much formality will overburden small projects. We believe, however, that informal approaches account, to some extent, for the poorer performance of small projects.

To achieve a sensible balance between overburdensome formality and unstructured informality, PEP members need to provide formal but flexible approaches and methods for small projects. Small maintenance projects, which have particular performance-inhibiting characteristics, need special attention, and small new development projects could often benefit from the use of modern methods and tools. Regardless of the approaches, methods or tools used, all small projects should be subjected to rigorous quality-assurance and risk-assessment procedures.

Provide formal but flexible approaches and methods

Most methods have been designed for use on large projects

Suppliers of methods have, by and large, designed their products and services for use on new (and by implication, large) development projects. User organisations have considered the available methods to be inappropriate, or even unnecessary, for maintenance and small projects. No supplier we met had any methods designed specifically for small projects, although some of them provide training in a 'rapid' version of their method, which may be relevant to small new development projects. However, one of the leading suppliers of methods in the United Kingdom, LBMS (the originators of SSADM, which has been adopted as a standard by government computing installations), expects to launch an update to its commercial version (now called System Engineer) in 1991. This new version will accommodate different types of projects, such as prototyping projects, new developments and re-engineering projects, as well as enhancement projects.

At present, however, the formal methods used by PEP members are usually most applicable to new, large developments; only a minority of members use formally defined methods for small projects. While 40 per cent of those we surveyed claim that they have different methods for small projects, or are in the process of preparing them, and 35 per cent claim that they differentiate between small new and small enhancement projects, the methods used for small projects are usually those that are also used for large projects. When such methods are used on small projects, some of the tasks will inevitably be carried out with less rigour. In most cases, it is left to project managers to decide what approach and method to use for a small project. While this has its merits, essential tasks can and do get overlooked.

Usually, the project manager decides on the method to be used for a small project

A few organisations have defined different approaches for different types of project, but these are often driven by technical factors (programming languages, for example) rather than by considerations of project size. In other cases, small projects have been deliberately overburdened by setting low size limits for the use of general approaches and methods, in order to make the development process more formal.

A minority of PEP members have recognised that a different approach is needed for small projects. Often, the emphasis is on reducing the number of major milestones. For example, The Co-operative Wholesale Society (a UK retail group) categorises projects according to their cost — those up to £5,000 have two checkpoints, those up to £50,000 have five checkpoints and those above £50,000 have eight checkpoints. Sun Alliance (a large insurance company) has three levels, with two, three or four main milestones and associated sign-offs, depending on the man-months of effort — the boundaries are at six man-months and 24 man-months.

Some organisations use a different approach for small projects

We believe that approaches and formal methods should be prescribed for all projects, including small projects. PEP members with at least 20 per cent of their endeavour in small projects, involving at least 20 staff, should define separate approaches depending on size and type, as well as on technical factors. This will narrow the range of options that may be considered at the outset of a project and should ensure that no essential tasks are overlooked. In defining the approaches, care should be taken to keep an appropriate balance between control, risk and performance.

PEP members considering acquiring a proprietary method should assess its suitability for different types of project. If alternative routes through the method are not predefined, the opportunities for tailoring it to suit different types of projects should be evaluated. Any proprietary method acquired should be able to support at least all the main types of projects undertaken, including small projects.

A proprietary method should also be able to support small projects

When deciding which approach and method to use for a particular project, consideration should also be given to any particular project characteristics that may warrant modifying the approach or method. This applies equally to small projects and to large ones. Any variations should be agreed at the outset of the project and recorded in the project plan.

Give small maintenance projects special attention

Many small projects are maintenance projects. The methods used for such projects usually follow the traditional informal approach to systems development, even when structured methods have been used to develop the original system. This is because of the tendency to maintain the lowest-level documentation — that is, the source programs themselves. We believe, however, that a predefined formal approach should be adopted for small maintenance projects, and that the methods selected for use by the department should be capable of being tailored for use on such projects. In particular, there should be a clear checkpoint between the problem-analysis and implementation stages.

PEP members should also seek to exploit tools that support the maintenance of existing projects, particularly those that help in batching amendments to regularly maintained applications and in controlling the release of new versions. They should also be aware of the implications for future small maintenance projects when selecting methods for new systems development.

Create a checkpoint between problem analysis and implementation

The lack of a checkpoint often results in poor technical quality and low productivity

There is a tendency with small maintenance projects to treat the analysis stage (often referred to as problem analysis) and the implementation stage as a continuous endeavour. This often results in poor technical quality and low productivity, because problems that should have been identified at the analysis stage do not show up until the implementation stage.

We believe that it is important to have a checkpoint at the end of the analysis stage of small maintenance projects. The purpose is to assess the quality of the analysis work and to identify and agree on the systems-testing requirements. At this checkpoint, it is also important to consider whether the maintenance project should proceed, or perhaps be combined with other projects for the sake of efficiency. In some cases, the checkpoint review might identify that a different approach would be appropriate, in the light of new perceptions of the size of the project or of the risk involved.

Batch amendments for regularly maintained applications

Testing can account for a large proportion of effort on small maintenance projects

The team or individuals working on some small maintenance projects may apparently be very productive, but their high productivity is often offset by poor technical quality. Testing can account for a large proportion of the effort used on small maintenance projects, and shortcuts are often taken to maintain the level of productivity.

One way of reducing the testing load on small maintenance projects is to batch changes so that the testing effort can be spread more economically. This is sensible provided that a change is not time-critical. If it is, the user should be expected to carry the extra cost of properly testing a small maintenance project. A further refinement is to batch changes so that new releases of an application can be released at fairly frequent intervals. For example, the application might be updated every six months with an agreed schedule of changes. Batching changes in this way is especially appropriate when several changes affect the same programs. This approach may also result in a more economic size of project, because several small projects can be combined in a single larger project.

Regardless of the approach adopted for testing small maintenance projects, performance may be improved by creating and maintaining a standard set of tests that the application must be able to process without error. Standard tests are particularly appropriate for frequently maintained applications.

Seek to exploit tools to support the maintenance of existing systems

Most tools are purchased because they can be used for new developments

Fewer than 30 per cent of the PEP members we surveyed in the research for this paper used special tools for supporting small projects. In most cases, a new tool is purchased because of its ability to support new developments. As a consequence, we believe that PEP members may not be taking advantage of the opportunities for supporting maintenance projects better, with tools such as:

- Automated programming-support environments (Maestro, for example).
- System-testing tools (see PEP Paper 13, *Software Testing*).
- Data dictionaries, for tracking systems data and supporting 'impact' analyses.
- Maintenance-support tools, such as code analysers, change-control tools and restructuring and re-engineering tools (see PEP Paper 8).

Consider the maintenance implications of selecting particular methods for new developments

New systems development methods should be chosen with a view to making subsequent maintenance as effective as possible. Many believe that structured methods have a role to play in this respect, although, in practice, this seldom happens.

At present, the code itself is often the only reliable form of documentation

The natural tendency for those engaged in maintenance work is to update only the lowest level of systems documentation, which is often the code itself. Higher levels of documentation are often neglected. Until CASE technology reaches the stage where it removes the need to maintain programs, the code itself will remain the only reliable form of documentation. For this reason, stringent documentation standards should be applied at the coding level. Even basic documentation standards, such as those for naming labels, can be absolutely critical to maintainability. As Figure 3.1 indicates, some of the world's best-known programming experts seem to favour documenting small units of code, limiting comments to explanations of special features and interfaces. Apart from this, they believe that an 'elegantly' structured and coded program requires no further documentation.

Figure 3.1 Views on documentation of some well known programming experts

Gary Kildall, founder and chairman of Digital Research: 'Properly written code is very much self-documented'.

C Wayne Ratcliff, originator of dBase: 'The more comments you need, the worse your program is'. He also tries to ensure that a program specification is not more than one page long.

Bob Frankston, originator of VisiCalc: 'The comments are there mainly to warn about surprises'.

Jonathan Sachs, writer of Lotus 1-2-3: 'I comment heavily on the description of the module and its inputs and outputs. There's no point in trying to document all the internal workings of each module.'

Application documentation should also include good high-level overviews and clear illustrations of the structure of the application, its programs and data, so that the units that need to be maintained can be identified easily. Modern techniques and methods do not automatically provide such high-level overviews. The design documentation produced by such methods may be too detailed and bulky, and staff may doubt its accuracy. High-level documentation is likely to change less often than low-level documentation, but it must be updated when changes affect it.

Staff need to be trained to apply documentation standards for small maintenance projects

Adhering to documentation standards for systems design, for program design and for the programs themselves is critical for subsequent maintenance efficiency. Updating the documentation should be mandatory for all maintenance projects; small maintenance projects are no exception. Staff therefore need to be properly trained to understand and apply the standards. They may learn the broad principles on external courses but will have to learn how to apply them in the development department itself. Training and support is essential at both levels.

Use modern methods and tools where appropriate on small new developments

A large proportion of the projects submitted to PEP still use third-generation languages and traditional or formally structured methods. We believe that, for small new developments, PEP members could make greater use of modern methods and tools. At present, the use of these may be constrained because they are not part of the systems department's strategic toolkit. In this situation, their use should be seriously considered when such strategies are under review.

Use modern languages and CASE for small new developments

Substantially higher levels of performance result from using modern languages

From our analyses of the projects submitted to PEP for assessment, it is clear that substantially higher levels of performance, expressed as function points per man-month, can be achieved by using modern languages. In practice, PEP members have found that some of these languages are suitable only for small new developments. One PEP member found this to be the case with Focus, another with CSP, and another with Ideal. PC developments, although not always small, can use a wide range of highly geared modern languages, such as dBase and DataEase. (High gearing means that fewer source statements are needed for each function point.) The delivery rates with these languages can be as high as 50 function points per man-month, compared with a typical rate of delivery of 10 function points per man-month for projects using third-generation languages.

Code-generation tools are causing projects to become smaller

Code-generation tools are becoming more popular, although again, their use is not limited to small projects. Nevertheless, the use of such tools is resulting in smaller projects in terms of manning, and the levels of productivity being achieved are frequently well above those achieved by projects using older languages.

Some claim that the use of integrated CASE tools results in high function-delivery rates. James Martin Associates (JMA), for example, claims that, with its CASE tool (IEF), it is possible to deliver between 30 and 50 function points per man-month (about two to three times the average for PEP projects). The use of integrated CASE not only has a direct impact on the productivity of code generation, but may lead to significant reductions of effort in other areas, too. JMA claims that systems-testing effort and errors are reduced because IEF automates the integration of system components. Such tools are also suitable for prototyping and iterative development approaches, which can also reduce development effort significantly. In PEP Paper 20, to be published at the end of 1991, we shall assess the impact that CASE is having on improving productivity.

Use rapid development approaches, especially where modern tools are available

Smaller projects are often based on rapid development or prototyping

Systems development experts increasingly advocate the use of 'rapid' and 'joint' application development approaches. These approaches are generally based on the reductions in time and effort that can be achieved by using CASE technology and fourth-generation languages. They often use a prototyping approach to identify and/or confirm the system requirements, and they are usually based on short, fixed time periods (often called 'time boxes') within which project objectives are to be accomplished. Keeping the size of projects down in this way will, in theory, avoid unnecessarily high rates of manpower buildup (and thus improve productivity), but this may not always be achieved. Although rapid development approaches can be used for a wide range of projects, we believe that their use is likely to be particularly suited to small projects, especially if the 'time boxing' concept is used.

Iterative development approaches are also more likely to be appropriate for small new developments. The availability of fourth-generation languages and CASE tools facilitates the use of these approaches. PEP members, however, seem to be making little use of iterative development.

Apply rigorous quality-assurance and risk-assessment techniques

Forty per cent of the PEP members we surveyed claim to have different quality-assurance practices for small and large projects, and 30 per cent claim to have different practices for enhancement and new projects. Often, though, the quality-assurance activities for small projects are a limited subset of the activities applied to larger projects. More often than not, it is left to the project manager to decide which quality-assurance activities to carry out for small projects. Moreover, if a quality-assurance group exists, it may not have the opportunity to challenge the project manager's decisions until after the event.

A few PEP members, however, follow the same quality-assurance procedures for small projects as they do for large ones, recognising that small projects may be relatively high-risk. We believe that all small projects should be subject to quality-assurance peer-group reviews, and to some measure of risk assessment.

Apply quality-assurance reviews to small projects

Any deviations from quality-assurance standards should be agreed before a small project begins

The project plan for a small project should identify any planned deviations from the usual quality-assurance standards, and these should be agreed before the project starts. Full-scale 'quality plans' are rarely applied to small projects, yet the lack of a quality plan can have as significant an impact on productivity and quality as it does for larger projects.

Peer-group review of planned changes to an existing system encourages more care

Peer-group reviews should also be applied as rigorously to small projects as to larger ones. Some PEP members feel that peer-group reviews are not appropriate for maintenance projects, because often only one person has an intimate understanding of the system. In our view, the need for peer-group reviews is probably more critical for maintenance projects. Maintaining applications software must not be an individual endeavour. Requiring the person who understands the system to explain how the change is intended to be implemented will encourage more care and will spread knowledge about the system. In a similar vein, one PEP member uses peer-group reviews at this crucial stage of a small maintenance project so that experienced people, who have moved into other areas of work, can scrutinise changes that have been defined by less experienced staff.

Changes to systems that were developed by others are often dealt with as 'enhancements' rather than as pure 'maintenance'. This often results in the changes being implemented as add-ons or 'fringe code', instead of being incorporated where they most logically fit in the structure, and thereby preserving the integrity of the original design. Each change therefore makes any future changes increasingly difficult and risky. Careful peer-group review of planned changes may avoid some of these problems.

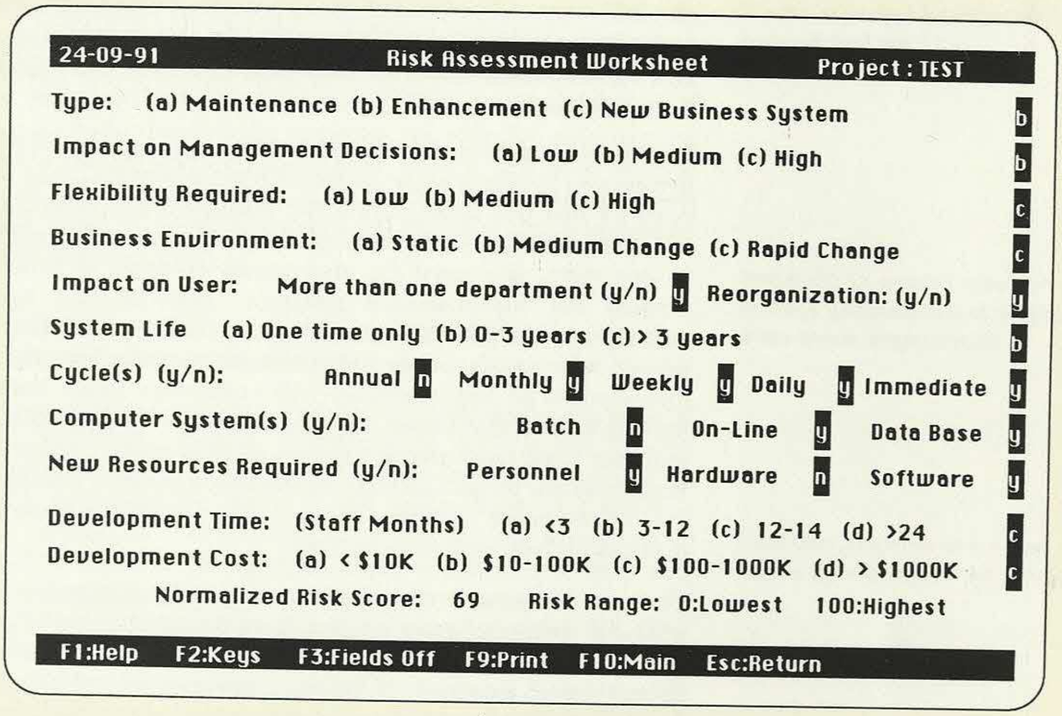
Develop risk-assessment procedures for all projects

A small minority of PEP members state that the level of risk involved influences their decisions about the approach and methods that they will use for small projects. The development standards and guidelines of some members specify the 'risk' factors that should be considered in deciding on the approach and methods that will be used. Abbey Life (a major insurance company), for instance, takes account of the availability of the required skills, the state of the existing documentation, the testing requirements, the need for measurement, dependencies on other systems, and other influences. Typically, however, the degree of risk tends to be considered in terms of technical factors and factors internal to the systems development function, rather than in business terms.

For each development approach used, PEP members should provide a clear statement of the types of risk and uncertainty for which that approach is appropriate and not appropriate. We also recommend the use of a simple rating mechanism (such as the one available with Hoskyns's Project BRIDGE and shown in Figure 3.2) for assessing risk and helping management to make quick and effective decisions about the approach and method that should be used for a particular project.

Figure 3.2 A simple rating mechanism will help managers to assess the level of risk involved in a project

With the risk-analysis function of Hoskyns's Project BRIDGE, risk is measured using such factors as cost of project, development time, system life and impact on user. The options selected are highlighted on the sample display shown below. The normalised risk score is then calculated on the basis of the selected options.



```
24-09-91                 Risk Assessment Worksheet              Project : TEST
Type:   (a) Maintenance  (b) Enhancement  (c) New Business System            b
Impact on Management Decisions:   (a) Low  (b) Medium  (c) High              b
Flexibility Required:   (a) Low  (b) Medium  (c) High                        c
Business Environment:   (a) Static  (b) Medium Change  (c) Rapid Change      c
Impact on User:   More than one department (y/n) y  Reorganization: (y/n)    y
System Life   (a) One time only  (b) 0-3 years  (c) > 3 years                b
Cycle(s) (y/n):        Annual n  Monthly y  Weekly y  Daily y  Immediate     y
Computer System(s) (y/n):          Batch  n   On-Line  y   Data Base         y
New Resources Required (y/n):  Personnel  y   Hardware n   Software          y
Development Time:  (Staff Months)  (a) <3  (b) 3-12  (c) 12-14  (d) >24     c
Development Cost:  (a) < $10K  (b) $10-100K  (c) $100-1000K  (d) > $1000K    c
         Normalized Risk Score:  69    Risk Range:  0:Lowest   100:Highest
F1:Help  F2:Keys  F3:Fields Off  F9:Print  F10:Main  Esc:Return
```

(Source: Hoskyns Group plc)

It is important, however, not to overburden small projects, and to match the rigour with which risk is assessed with the type of undertaking. A two-stage approach is likely to be appropriate for small projects: a simple, initial assessment to estimate the level of risk and select the approach, followed by a more elaborate assessment, using more formal approaches, for projects that have been identified as potentially high-risk. When the project has been completed, the risk rating should be re-examined to assess whether the most appropriate approach was taken and to learn from the outcome.

A two-stage approach to risk assessment for small projects is usually appropriate

In this chapter, we have considered how PEP members might improve the performance of small projects by paying more attention to the methods that they use. In the next chapter, we consider ways in which the performance of small projects might be further improved if appropriate organisational arrangements are made for managing them.

Chapter 4

Organise and manage small projects separately

PEP members are organised to undertake small projects in a variety of ways. Many carry out small project work alongside other projects. A few, however, have a separate unit dedicated to small project work. These two basic options may be complicated by the organisational structures in place for large projects, such as splitting the development department by development stage, or using matrix management of a pool of development resources, or some combination of these. These various organisational structures were discussed in detail in PEP Paper 11, *Organising the Systems Development Department.*

In our view, the performance of staff working on small projects is maximised by creating a separate unit dedicated to small projects. However, special controls are needed to manage small projects properly, and special consideration needs to be given to the unique human factors involved, such as the increased sensitivity to individual performance.

Establish separate units for small projects

We recommend that PEP members set up separate organisational units to undertake maintenance projects, and where possible, also create a special unit to handle 'rapid' new systems developments. Organisational arrangements should not, however, remain static. As experience is gained, boundaries between organisational units may need to be adjusted, and organisational units may need to be formed or dissolved as the mix of work changes. It is also important to ensure that the number of projects being undertaken by any one individual should be kept to no more than three.

The boundaries between organisational units should be reviewed as circumstances change

Set up a separate organisational unit for maintenance projects

Small projects are often maintenance projects for existing applications. The case for separating maintenance work from other development work has been discussed previously in PEP Papers 8 and 11, although only a minority of PEP members have adopted this approach. Among the benefits cited by those who have adopted the approach are:

- It raises the profile of small projects and ensures that more management attention is given to them.
- It provides users with a more responsive service.
- It enables teams working on new developments to work without distraction.

Incomplete documentation hinders effective organisation for maintenance

The major problem cited by those who have not adopted the approach is the handover of applications from the teams that originally developed them to maintenance staff. This problem is particularly acute when the original developers do not take account of the long-term implications of their design and documentation decisions. The subsequent maintenance difficulties mean that more effective organisational options for maintenance cannot be considered. However, if design and documentation standards are set and enforced, and if quality-assurance procedures are consistently practised, there should be no problem in handing over applications to a dedicated maintenance unit.

In 1989, Abbey Life changed its development organisation so that small maintenance projects would receive much more management attention. Its new structure and the benefits are described in Figure 4.1.

Figure 4.1 Abbey Life has created a special organisational structure for small projects

In the past, the systems department at Abbey Life consisted of teams that carried out both new project and maintenance work. In the current structure, which has been in place for about two years, small projects are dealt with separately. One development group (financial systems) illustrates the approach. The group, managed by an account manager, is split into three units:

— The projects unit, consisting of about nine staff, undertakes all projects of 12 or more man-months, or smaller ones if they are judged to be critical.

— The systems-support unit, consisting of about eight staff, deals with all development-activity requests (DARs) that lead to projects of less than 12 man-months. A user-led priorities group, composed of representatives of middle and junior management, meets monthly to agree on the priorities for and the scheduling of the requests. If a DAR expands to become a project of 12 or more man-months, it is passed on to the projects unit.

— The essential-support unit, consisting of three staff, deals with problems in operational systems. Much of its work is concerned with fixing problems or omissions caused by recent changes. This unit deals with all fix-or-fail and top-priority DARs, which normally originate from the user and operations areas. Staff assigned to the essential-support unit work in the systems-support unit if they have no high-priority DARs in hand. The head of this unit decides on the sequence in which the DARs are acted upon.

Staff are continually rotated through the two support units and are also allocated to projects. A planning group of business directors decides on the overall distribution of resources to the organisational units, and on the projects that are to be undertaken. Staff can be assigned to work on several small projects at any one time.

The new structure has improved the management of small projects, although there are still some problems to resolve:

— The detailed understanding of applications by staff previously involved in supporting them has been diluted, because many of the so-called 'gurus' of existing systems have been located in other development units.

— The deficiencies in existing systems documentation and testing facilities have been exposed.

— A program may be the subject of several maintenance projects, which can lead to difficulties with parallel development. Abbey Life tries to minimise the occasions on which this can occur, and will reschedule non-essential tasks to prevent it from happening.

— It is easier to overlook some essential activities on small projects as a result of trying to be less formal.

Abbey Life believes that its small projects perform better than they would have done with the previous structure. The structure is, however, currently being reviewed to ensure maximum efficiency, in the light of the experience gained during the last two years.

Create a special unit to handle 'rapid' new developments

Although the focus of small projects is often on maintenance, small new developments must not be ignored. If such projects are likely to require a significant number of development staff— say, at least 10 people — consideration should be given to creating a separate group to be responsible for them, too.

'Rapid' approaches to systems development require some special skills, such as those needed to run workshops and similar working groups. We believe that these skills are best concentrated in an organisational unit separate from those responsible for other types of development. In some organisations, information centres may provide some small new development capability, although they usually concentrate their efforts on providing technical advice (for example, the use of PCs and the use of information-retrieval tools on mainframes).

The special skills required for rapid development should be concentrated in a separate unit

Minimise the number of projects being undertaken by any one individual

The most usual case is that one member of staff is assigned to each of the small projects being undertaken by PEP members and he works only on that project. However, the actual manning ranges from five or six staff dedicated to one small project to one member of staff responsible for seven small projects (see Figure 4.2).

Figure 4.2 Manning levels on small projects vary markedly

Persons per project	Number of organisations
Five or six	5
Three or four	5
One or two	18
Less than one (that is, more than one project per person)	14

(Source: Survey of 42 PEP members)

While, in theory, it would be ideal to assign only one project per person at any one time, this would often result in inefficient use of overall resources, and 'over-manning' of small projects. It is difficult to work full-time on one small project — there will inevitably be delays while waiting to meet a user, or waiting for some related external piece of work. In reality, three small projects per person is likely to be the most efficient allocation of resources.

Three small projects per individual is likely to make the most efficient use of resources

Provide unique management controls

Few PEP members have formal project-management methods in place for small projects. About one-third of the members we surveyed said they used different methods for small projects; these tended to be the larger organisations. Another third claimed to use different project-management methods for small new and small enhancement projects; these tended to be the smaller organisations. For most organisations, however, the project-management methods used for small projects were a subset of those used for larger projects. For example, control documents were simpler, and formal reviews by management, if done at all, were done at a less detailed level.

Usually, the project-management methods for small projects are a subset of those for larger projects

There are five features of the control of small projects that do need special attention:

— Balancing the controls with the size of projects, and the risks inherent in them, to avoid overburdening a small project.

— Establishing mechanisms for setting and reviewing priorities for small projects to avoid arbitrary decision-making.

— Identifying the factors that could inhibit the progress of small projects before they begin, to minimise the need for constant rescheduling.

— Setting budgets for levels of maintenance according to user/business needs.

— Establishing clear acceptance criteria for handing over new systems to maintenance teams.

Balance controls against project size and risk

While it is realistic for small-project controls to be generally less stringent than those for larger projects, some small projects may be both critical and risky, and they should therefore be subject to additional controls. It is important, however, not to over-exaggerate the risk of a small project, because this will result in excessive controls being imposed, and in poorer performance. There is also a danger that the project manager will have too much discretion in deciding on which project controls to impose, and this, too, can result in poor performance (and in high levels of rework). A balance must therefore be struck between the level of project control imposed on a small project and the risk of that project's failing.

Over-exaggerating the risk of a small project will result in excessive controls

It is sensible to keep management-control mechanisms and the frequency and type of reporting as simple as possible. The approaches and methods applicable to small projects should include guidelines on the levels of control that are likely to be appropriate. For example, neither a full steering committee nor a single 'steering' sponsor may be appropriate for a small project. A scaled-down steering committee may be the best alternative. When each small project is initiated, these aspects should be reviewed, and the arrangements agreed should be documented.

Establish mechanisms for setting and reviewing small-project priorities

Various approaches are used by PEP members to assign priorities to small projects. It may be the responsibility of individual users, of development managers, or of a priority or steering committee, usually depending on who is paying for the work. If a user department is paying and the development department has a fixed level of resources available for that department's projects, the user should determine the priorities for small projects. When corporately funded resources have to be divided among many users, the mechanisms for deciding on priorities need to be clearly defined.

One of the problems that particularly besets small projects is that of changing priorities, especially where they are originally set according to 'who shouted the loudest'. Development managers often consider that it is less disruptive to switch priorities on small projects, but the impact on development staff can be quite unsettling. Work may have to be abandoned and re-done at a later date, thus wasting scarce resources. In the meantime, staff and projects are re-assigned and schedules have to be reworked. PEP members should seek to ensure that priorities for small projects are set as objectively as possible and that changes to them are minimised.

Changing the priorities on small projects can be disruptive for development staff

Identify the factors that could inhibit progress

Individual small projects are more sensitive to delays and dependencies on other systems than larger projects, where a delay in one area does not usually cause the whole project to come to a standstill. On small projects, it is therefore wise to consider, at the outset, and in detail, the dependencies and potential delays to which they could be subjected, and to make allowance for these in project planning. Continually stopping and starting a small project can have an adverse effect on performance, so it may be sensible to delay the start of a small project until the dependencies and potential delays are removed, or minimised.

Set budgets for levels of maintenance

PEP members who have separate maintenance units often have specific budgets for maintenance work, to ensure that such work (particularly small maintenance projects) is not squeezed out by demands for new developments. The budget level is usually set annually, and thereafter, individual projects are identified and priorities are set to use the available resources. This approach is sensible, because it means that management does not need to make a large number of decisions about a constant stream of small maintenance projects. In organisations where the level of maintenance is very variable, it may be more difficult to assign a fixed level of resources to maintenance work. Doing so will, however, give a clearer focus to small maintenance work and will make it more effective.

Fixing the level of maintenance resources will make small maintenance projects more effective

Establish clear acceptance criteria for handing over new systems

We have already noted that the performance of maintenance projects is significantly affected by the type and condition of the existing documentation. To ensure that a separate maintenance unit is as effective as possible, PEP members should therefore establish clear criteria for new applications being handed over to support teams. These criteria should specify both the minimum error level that applications should have reached (in terms of frequency of error), and the levels and type of documentation required. Failure to meet the criteria should mean that the original teams carry the penalty of either rectifying any problems or, perhaps, bearing the cost of rectifying them.

Manage the unique human aspects of small projects

The performance of a small project can be significantly affected by the behaviour and idiosyncrasies of individuals. It is therefore very important that project managers adopt a management style that is appropriate for small-scale endeavours, and that staff are not assigned without considering both their suitability for such work and their ability to work well as part of a small group.

Ensure that the style of the project manager is appropriate for small projects

Experienced project managers may not be the best choice for small projects

PEP members need to take special care in selecting project managers for small projects. In many cases, experienced project managers, accustomed to working on large projects where a formal, directive management style is appropriate, will not be the best choice. Less experienced managers, who can act as 'leader' and influence team members by coaching and gaining their consent while working alongside them, will often be more effective on small-project work.

Ensure that staff are suited to small projects

Assigning staff to small projects and allocating the various aspects of the work involved to particular people is not a simple task. Two points of view have to be taken into account — that of the individual, and that of the development manager.

The shorter timescales can make it attractive to work on a small project

From the individual's point of view, a small project may be attractive because the shorter timescales can provide the opportunity to work on a complete development from beginning to end. The benefits from a small project are available earlier to the users, who may therefore be more supportive, and give the development staff greater recognition. A particular individual's attitude to small-project work may, however, vary over time. The attitudes of staff towards small projects need to be monitored, and where possible, their changing needs should be accommodated.

Perceiving small maintenance projects as a means of providing on-the-job training may be counter-productive

From the development manager's point of view, small, less risky projects may provide opportunities for on-the-job training. Yet, for maintenance work, it may be more important to assign programmers who are very knowledgeable about the existing code. Systems development managers need to identify the factors that are most critical in each circumstance, and select staff accordingly. Such considerations should form an integral part of the formal risk assessment associated with small projects.

Another important consideration in staffing small projects is the fit between the individual and the work. Small projects are often staffed by no more than one or two people, so tasks need to be aligned more to an individual's particular abilities. Furthermore, the work itself is not as readily partitioned, particularly the problem-analysis stage in maintenance. Staff working on small projects should be encouraged to think more about the deliverables required than the methods they use, and project managers need to steer and counsel them, rather than direct them.

Ensure that team members are working well with each other

We saw in Chapter 2 that the influence of an individual on the performance of a small project can be quite profound. It is therefore very important for development and project managers to be aware of small-team dynamics and to monitor them carefully.

Personality characteristics are more critical on small projects

A non-conformist or a loner can be tolerated in large projects, but in a small team, the influence of such people is more marked and project performance is affected accordingly. An effective small team can therefore be more difficult to create. Both the extremes (an excessively close-knit group and a loose collection of individuals) are common. In PEP Paper 7, *Influence on Productivity of Staff Personality and Team Working*, we identified some of the characteristics of IT people that can undermine effective team working. We also identified the need to ensure a balance both of personalities and of skills within a team. PEP members should seek to ensure that the natural tendency of development staff to be 'thinking' types of people is balanced by at least some 'feeling' types. Particularly successful combinations of individuals in small teams should be noted so that the same combination can be used in the future. Loners, however, should be given self-contained tasks or projects.

Chapter 5

Improve the performance of small projects

We have seen in this paper that although there are wide variations in the performance of small projects among PEP members, they generally perform less well than large ones. Figure 5.1 lists the actions that systems development managers should take to ensure that the overall performance of their department is not adversely affected by the performance of small projects, which absorb a large and growing proportion of scarce systems resources.

Figure 5.1 Action checklist

Review current measurement practices and determine:

— How these should be enhanced to collect measures systematically and accurately for all projects, including small ones.

— How analyses of measurements might be improved to increase managers' awareness of performance – particularly for small projects – and the quality of their decision-making.

— How program blackspots, which cause some small maintenance projects to be very inefficient, can be identified and monitored as a basis for identifying the need for selective redevelopment.

— How measurements can be enhanced and used to estimate small projects more accurately, particularly small maintenance projects.

Review existing methods and the extent to which they support small projects. In particular:

— Plan to introduce, where necessary, formal methods suited to the different types of small project carried out.

— Develop practices within the methods for assessing project risk and selecting the most appropriate approach.

— Provide mechanisms that encourage changes to be batched for maintenance projects, and introduce releases for frequently maintained applications.

— Evaluate the suitability of 'rapid' approaches for small new developments.

Review program and documentation standards and ensure that they concentrate on the critical elements – namely, the source programs themselves and overviews of the high-level design.

Examine quality-assurance practices:

— Revise these practices where necessary to ensure that documentation standards are strictly adhered to.

— Ensure that these practices address the needs of small projects, and at least adopt peer-group reviews of design specifications.

Review the planning procedures for small projects and ensure that they are thorough.

Review the present organisational arrangements, particularly as they apply to small projects:

— Consider setting up separate unit(s) for small maintenance projects.

— Consider setting up a separate unit for small new developments, especially if they are to be based on 'rapid' development approaches.

The main reason for the discrepancy in performance is that while small projects have many features that distinguish them from large ones, they tend to be treated in the same way. Systems managers who continue to organise and manage small projects as if they were simply downsized large ones should not be surprised if they fail to bring about any improvement in their performance.

BUTLER COX
P.E.P

Butler Cox

Butler Cox is an independent, international consulting company specialising in areas relating to information technology.

The company offers a unique blend of high-level commercial perspective and in-depth technical expertise, a capability which in recent years has been put to the service of many of the world's largest and most successful organisations.

Butler Cox provides a range of consulting services both to organisations that are major users of information technology and to suppliers of information technology products.

Consulting for Users
Supporting clients in establishing the right opportunities for the use of information technology, selecting appropriate equipment and software, and managing its introduction and development.

Consulting for Suppliers
Supporting major information technology and telecommunications suppliers in assessing opportunities, formulating market strategies, and completing acquisitions and mergers.

Foundation
The Foundation is a service for senior managers responsible for information management in major enterprises. It provides insights and guidance to help them to manage information systems and technology more effectively for the benefit of their organisation.

Education
The Cranfield IT Institute, a wholly owned subsidiary of the Butler Cox Group, educates systems specialists, IT managers, line managers, and professionals to understand more fully how to apply and use today's technology.

PEP

The Butler Cox Productivity Enhancement Programme (PEP) is a participative service whose goal is to improve productivity in application systems development.

It provides practical help to systems development managers and identifies the specific problems that prevent them from using their development resources effectively. At the same time, the programme keeps these managers abreast of the latest thinking and experience of experts and practitioners in the field.

The programme consists of individual guidance for each subscriber in the form of a productivity assessment, and also publications and forum meetings common to all subscribers.

Productivity Assessment
Each subscribing organisation receives a confidential management assessment of its systems development productivity. The assessment is based on a comparison of key development data from selected subscriber projects against a large comprehensive database. It is presented in a detailed report and subscribers are briefed at a meeting with Butler Cox specialists.

Meetings
Each quarterly PEP forum meeting focuses on the issues highlighted in the previous PEP Paper. The meetings give participants the opportunity to discuss the topic in detail and to exchange views with managers from other member organisations.

***PEP Paper*s**
Four PEP Papers are produced each year. They concentrate on specific aspects of system development productivity and offer practical advice based on recent research and experience. The topics are selected to reflect the concerns of the members while maintaining a balance between management and technical issues.

Previous PEP Papers
4 Requirements Definition: The Key to System Development Productivity
5 Managing Productivity in Systems Development
6 Managing Contemporary System Development Methods
7 Influence on Productivity of Staff Personality and Team Working
8 Managing Software Maintenance
9 Quality Assurance in Systems Development
10 Making Effective Use of Modern Development Tools
11 Organising the Systems Development Department
12 Trends in Systems Development Among PEP Members
13 Software Testing
14 Software Quality Measurement
15 Application Packages
16 Project Estimating
17 Motivating Systems Development Staff
18 Managing Small Projects

Forthcoming PEP Papers
Involving Users in Systems Development
The Impact of CASE

BUTLER COX

Butler Cox plc
Butler Cox House, 12 Bloomsbury Square,
London WC1A 2LL, England
☎ (071) 831 0101, Telex 8813717 BUTCOX G
Fax (071) 831 6250

Belgium and the Netherlands
Butler Cox Benelux bv
Prins Hendriklaan 52
1075 BE Amsterdam, The Netherlands
☎ (020) 6 75 51 11, Fax (020) 6 75 53 31

France
Butler Cox SARL
Tour Akzo, 164 Rue Ambroise Croizat,
93204 St Denis-Cédex 1, France
☎ (1) 48.20.61.64, Télécopieur (1) 48.20.72.58

Germany, Austria and Switzerland
Butler Cox GmbH
Richard-Wagner-Str. 13, 8000 München 2, Germany
☎ (089) 5 23 40 01, Fax (089) 5 23 35 15

Australia, New Zealand and South-east Asia
Mr J Cooper
Butler Cox Foundation
Level 10, 70 Pitt Street, Sydney, NSW 2000, Australia
☎ (02) 223 6922, Fax (02) 223 6997

Finland
TT-Innovation Oy
Sinikalliontie 5, 02630 Espoo, Finland
☎ (90) 358 0502 731, Fax (90) 358 05022 682

Ireland
SD Consulting
8 Clanwilliam Square, Dublin 2, Ireland
☎ (01) 764701, Fax (01) 767945

Italy
RSO SpA
Via Leopardi 1, 20123 Milano, Italy
☎ (02) 720 00 583, Fax (02) 86 45 07 20

Scandinavia
Butler Cox Foundation Scandinavia AB
Jungfrudansen 21, Box 4040, 171 04 Solna, Sweden
☎ (08) 705 83 60, Fax (08) 730 15 67

Spain and Portugal
T Network SA
Núñez Morgado 3-6°b, 28036 Madrid, Spain
☎ (91) 733 9866, Fax (91) 733 9910