Requirements Definition: The Key to System Development Productivity

BUTLER COX
P.E.P

Position Paper 4, November 1987

BUTLER COX
P.E.P

Requirements Definition: The Key to System Development Productivity

Position Paper 4, November 1987
by Mary Cockcroft

Mary Cockcroft is the manager of systems management consultancy at Butler Cox.

Since joining Butler Cox in 1984 she has worked on many consultancy and research projects in the systems development area. She has managed several consultancy assignments providing advice on system development project management, methods, techniques and tools and how to assure quality, including:

- Auditing a multi-million pound integrated administration and process control system, and advising on a range of management and technical issues associated with introducing the system.
- Recommending the system development methods and tools to be used by a multinational manufacturing company.
- Implementing quality assurance management for a multinational oil company.
- Developing a technical architecture for a large European-based organisation.

She is also the author of the Butler Cox Foundation report on 'Using System Development Methods' and was one of the researchers who contributed to the report on 'Building Quality Systems'.

She began her career in consultancy when she joined McKinsey and Company as a Business Analyst. She then joined the Systems Business Division of Rank Xerox Limited working on the analysis and development of marketing, planning, and financial systems.

In 1981 she joined Standard Telephones and Cables plc as a Business Information Analyst in Systems Planning and Control. She was a member of the team that developed the information management structure of the company using structured methods. During this period she was involved in constructing software tools to assist in the development process.

BUTLER COX

Published by Butler Cox & Partners Limited
Butler Cox House
12 Bloomsbury Square
London WC1A 2LL
England

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

BUTLER COX

P.E.P

Requirements Definition: The Key to System Development Productivity

Position Paper 4, November 1987

Contents

BUTLER COX

Chapter 1

Proprietary methods provide benefits: but requirements definition is the key to productivity

It is clear to system development professionals that, despite the significant advances achieved over the last decade, there are still many systems which are developed late and over budget. (The survey for the PEP Position Paper 3: 'Planning and Managing Systems Development' revealed more than half the projects had overruns of at least 25 per cent.) Moreover, it is still typical for users to complain that the systems they are delivered do not meet their real business needs.

Proprietary system development methods such as Information Engineering or Method 1 claim to offer the solutions to the above problems. In preparing this position paper, Butler Cox conducted a brief telephone survey of PEP members. They use a wide variety of methods (see Figure 1.1) but in the United Kingdom LSDM and Information Engineering are the most popular. The new factor which emerged during our research is the decline in importance of in-house methods. Typically these are now used to support and complement proprietary methods rather than act as a substitute for them.

Most companies use a wide variety of approaches to developing systems; all the companies contacted used the conventional 'lifecycle' and small systems approaches, and all bought in application packages. Use of the iterative approach was, however, more variable. Of the 22 companies contacted, two did not use it at all and the majority of the others only used it for 'screen painting' at the front end of development. There still appears to be much uncertainty about how best to implement the iterative approach.

Methods can improve productivity in terms of quality, cost, and timeliness during development if used effectively. However, the biggest improvement in productivity is achieved by defining the business requirements correctly at the very beginning of the development process. Our review of members showed, however, that despite the relatively large amount of time and cost they spent on requirements definition and analysis, the approaches they used could be greatly improved. It is in this area that the proprietary methods offer least support.

PRODUCTIVITY BENEFITS FROM SYSTEM DEVELOPMENT METHODS

It is widely accepted that proprietary methods provide two kinds of benefit:

- Improved quality of systems.
- Improved control over the development process itself in terms of cost and time.

Because of the well-known lack of measurement in system development, it is in fact difficult to identify the precise impact of the method on either the project itself or indeed on the resultant application system. However, recent work carried out by Butler Cox on system development methods, showed that these benefits could be achieved through seven features of the method:

- The visible plans and progress produced by formalising the development process.
- The use of standard checklists, tailored on a project by project basis.
- The use of standard techniques, in which all the development team are well trained.
- The use of graphics to present information to users and to other developers.
- Advanced requirements analysis techniques, such as the increased use of prototyping.
- The use of structured analysis and design techniques like data modelling, activity decomposition and entity life history.
- The use of structured programming and fourth-generation languages in system building.

These factors not only help in controlling cost and duration of the project but improve the system quality through:

- Ensuring completeness/correctness at each stage of the project.
- Improving communication between developer and user and between developers.
- Providing a basis for the automation of the analysis, design, and building of the system itself.

The use of a method does not, in itself, reduce the development time or cost — indeed it may add to them.

How each feature affects productivity is shown in Figure 1.2.

REQUIREMENTS DEFINITION IS CRUCIAL FOR EFFECTIVE DEVELOPMENT

It is critical for any application that its development starts on the right foot. There should be a business justification for the development, the benefits, the costs and risks (both business and technical) involved should be clearly identified, the boundaries of the proposed application should be clearly defined, and the requirements of the proposed application should be clearly understood.

The need for business justification, clear boundaries, a statement of benefits, costs, and even risks is accepted widely amongst system developers. However, the need for a very clear requirement specification at the front end of system development is perhaps not so clear. Particularly since the advent of fourth-generation languages and prototyping, there is a view that it is possible to manage with just a broad 'feel' for requirements and to work out the detailed requirements at a later stage

Figure 1.1 Use of proprietary system development methods by PEP participants contacted during survey

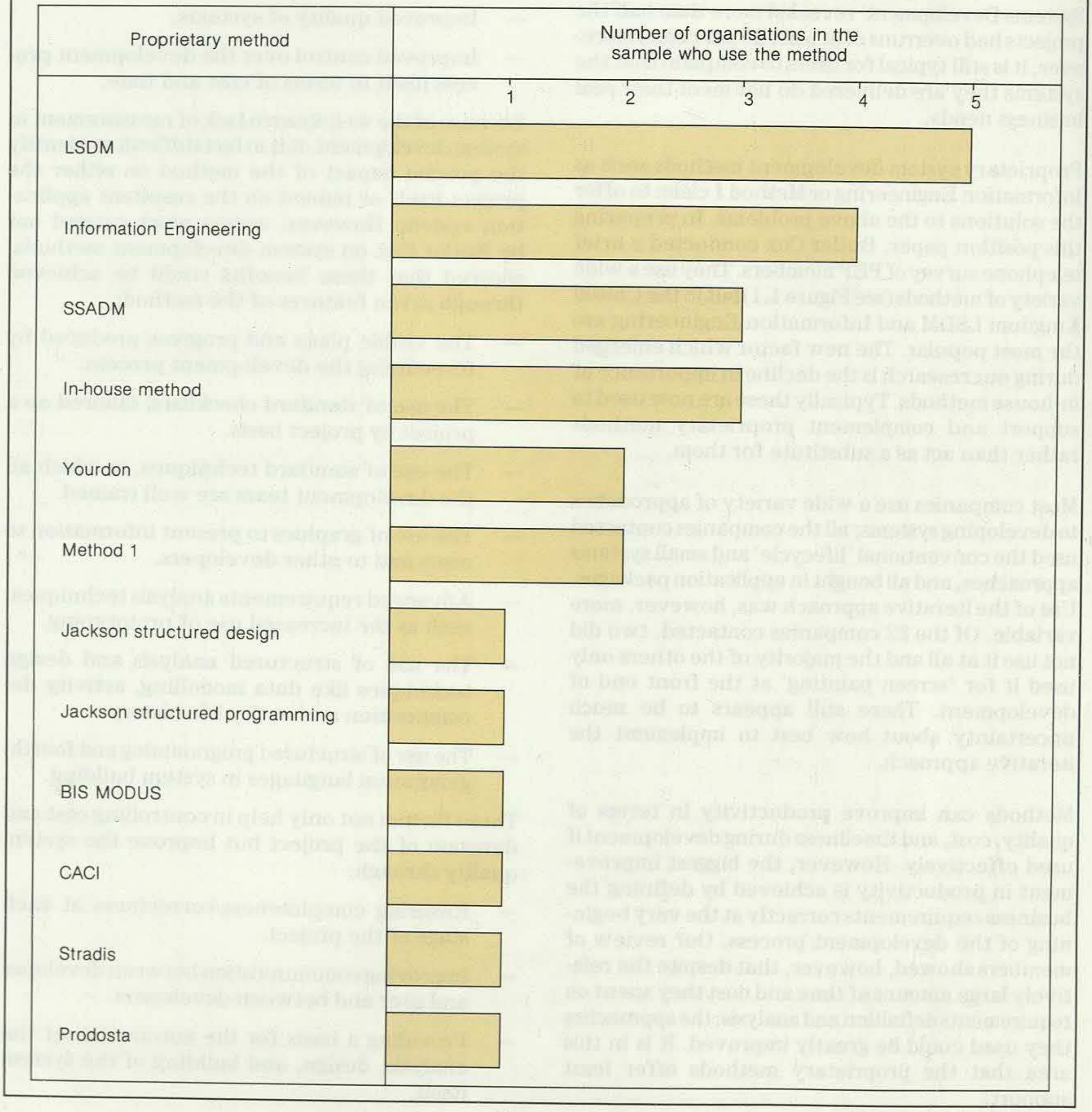


Figure 1.2 How methods can improve productivity

Impact on development / Feature of methods	Improves control over cost and time	Improves quality		
		Ensures completeness/ correctness	Improves communications with users and other developers	Aids automation
Visible plans/progress	✓	✓	✓	
Standard checklists and techniques	✓	✓	✓	
Graphics		✓	✓	
Advanced requirements analysis techniques (eg prototyping)		✓	✓	
Structured analysis/ design techniques		✓	✓	✓
Structured programming techniques		✓	✓	✓

Key:

✓ Feature has impact shown

in the development process. Often users are unclear about their real requirements at the start of development, and developers feel that they can 'guestimate' what they are likely to be, and adjust the system to match real needs during later stages of development. Sometimes, the business is changing rapidly and developers take the view that it is impossible for the requirements to be specified fully at the start and by the time the development project proceeds, the rate of business change will slow down and the requirements will be more clear. Thus, system developers may question whether it is necessary to define business requirements right at the start.

We believe that it is critical to do so right from the start of development for two main reasons. The first is that adding to or changing requirements at later stages in development costs incrementally more as development proceeds. The second is that without a realistic requirement specification, agreed with the user at the start of development, the users themselves are unlikely to be satisfied with the finished product.

The fact that adding to or changing requirements in later stages of development costs incrementally more as development progresses is borne out by the experience of most development groups, but is particularly evident when commissioning software from an external software house. Many organisations have been surprised to find that the quotations for development rapidly escalate, either as more requirements are identified or as the initial requirement specification is modified. Software houses have excellent change (scope) control mechanisms for this very reason. Boehm (an American software expert) estimates that a change costing $1 at the requirements definition stage will cost $10 in systems analysis, $100 in systems design, or $1,000 in testing or implementation if the change is made later. An analogy can be drawn with, say, automobile design and manufacture. Clearly, it is much less costly to modify the design at the drawing board stage than when the production line has been tooled up.

Users are unlikely to be satisfied with the finished product without a clear definition of requirements since no one (not even themselves) probably has a clear view of what will satisfy them, and without some sort of expectation management, success will be difficult to achieve. A common cause of friction between developers and users arises from the developers not recognising the need to manage the users' expectations.

Prototyping is, of course, a slightly different case. However, even when using prototyping, it must be clear that the front-end specification is merely that — a specification of requirements. The fact that it was developed by a technical process rather than by old fashioned personal interviewing using pen and paper is neither here nor there. What is beneficial with prototyping, however, is that it is possible to use the output of the requirements definition phase, that is the code and the associated databases (if they are available) as the basis for future technical development work.

IDEALLY, REQUIREMENTS DEFINITION SHOULD FOLLOW FROM A STRATEGIC PLAN

As part of the survey carried out for this paper, we asked PEP participants if they had a strategic analysis of business requirements available as a basis for initially assessing requirements during application development.

Only 10 per cent of the respondents (Figure 1.3) had a top-down requirements analysis of corporate needs available and were actively developing applications in line with it. Most companies were either in the process of developing their strategic analysis, or had not yet started developing it. In these latter cases, each application system developed was looked on as a separate project. This response is typical of both PEP participants as a whole and indeed of the more advanced European application development groups that we have observed.

This means that for many development groups, it is even more vital that they concentrate effort on the requirements definition in each application. Unlike strategic analysis, the business requirements definition phase of application development is under the control of the development group. Considerable effort and time *is* put into this phase of development by most companies (see Figures 1.4 to 1.6). There have been a number of innovations in requirements definition in the last few years, and it is increasingly recognised that there are many lessons to be learnt from other disciplines, particularly the techniques of market research. But our survey of PEP members suggests that there is much scope for improving the effectiveness of the effort they deploy on requirements analysis.

Figure 1.3 The use of strategic analysis as a basis for a system development

Only 10 per cent of the organisations surveyed use a strategic plan as the basis for system development

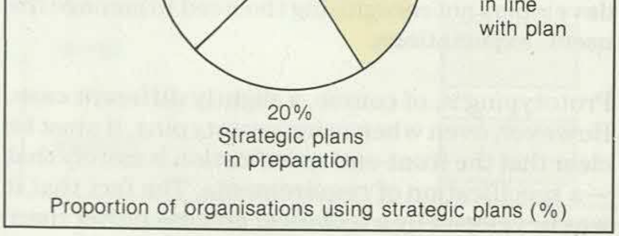


Figure 1.4 Expenditure on requirements definition

Some organisations spend more than 20 per cent of the development cost on requirements definition

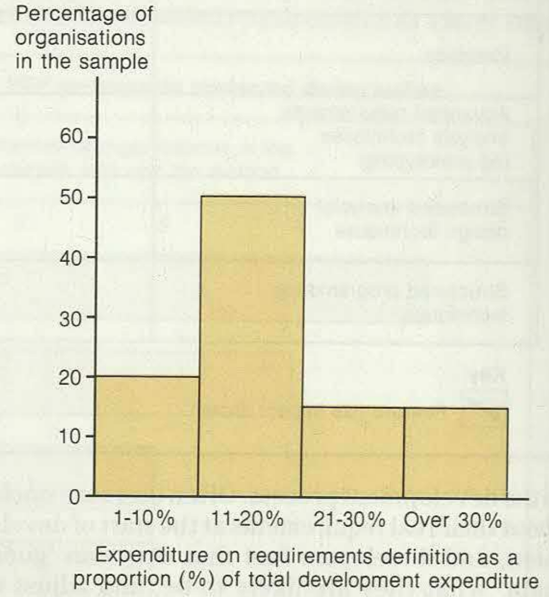


Figure 1.5 Effort expended on requirements definition

Some organisations spend more than 20 per cent of the development cost on requirements definition

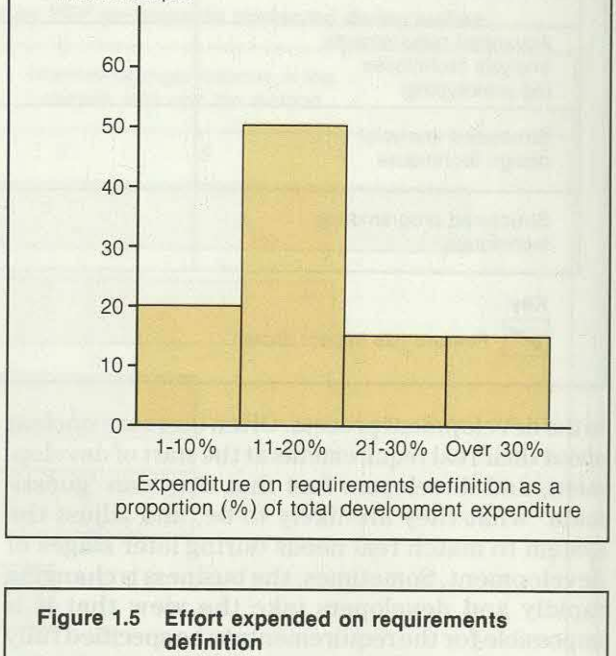
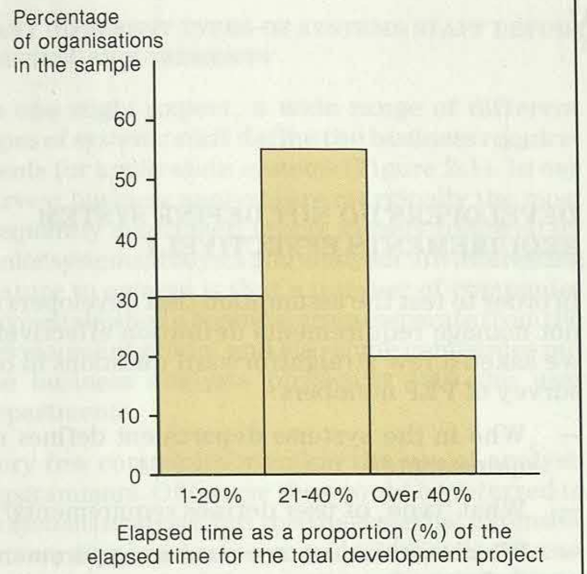


Figure 1.6 Elapsed time spent on requirements definition

Some organisations spend more than 40 per cent of the elapsed time in development on requirements definition



Chapter 2

Requirements definition is not managed effectively

There is a tried and tested body of knowledge on the effective collection of information about the needs of others. Professional market researchers, for example, have developed various techniques and tools to do just this. There is an analogy between the market researcher seeking to define the characteristics of a product that the customer will buy, and the system developer seeking to define the system the users will adopt. Based on our practical experience of working in and with a number of companies, discussing the subject with many system developers, and a review of the available development literature, it is our belief that this knowledge is not effectively used by system developers. They tend to concentrate on the technically interesting aspects of system development — which is their particular area of expertise, at the expense of the more mundane fact gathering.

A good example of this was a recent statement by a senior systems manager: "The requirements just seem to sort of . . . appear."

Naturally, the analysis, design, programming, and implementation of systems are critical. Those are the areas which make systems unique. However, if all this extensive effort is based on poor information, the old computing maxim of 'garbage in . . . garbage out' becomes a reality.

Although the sample of companies contacted during the research for this paper is small, the results of the survey suggest that developers do not research business requirements effectively. It is in fact very difficult in system development to identify what precisely did go 'wrong' or 'right' in any particular development, since a good requirements specification can be poorly implemented and a 'poor' specification can be vastly improved (at a cost) by analysts, designers, and programmers at later stages of development. However, the general opinion amongst the group was that there was room for improvement in requirements definition.

We also briefly reviewed the research techniques suggested by widely used proprietary development methods only to find that they too offer little guidance in requirements definition.

DEVELOPERS DO NOT DEFINE SYSTEM REQUIREMENTS EFFECTIVELY

In order to test the assumption that developers do not manage requirements definition effectively, we asked a few straightforward questions in our survey of PEP members:

- Who in the systems department defines requirements?
- What 'type' of user defines requirements?
- Which techniques are used in requirements definition?
- How do you decide which techniques to use?
- What experiences (good or bad) have there been with these techniques in your organisation?

Our major findings are:

- Requirements definition is seen mainly as an organisational issue by some developers.
- Many different types of staff define business requirements.
- Few development groups prepare formal research plans.
- Development groups seek information from a limited number of sources.
- Many of the available information collection techniques are not used.
- The use of prototyping is limited.
- Not all organisations use quality assurance techniques in requirements definition.

REQUIREMENTS DEFINITION IS SEEN AS MAINLY AN 'ORGANISATIONAL' ISSUE BY SOME DEVELOPERS

An unexpected outcome of the research was that when asked what techniques they use to identify requirements, one-third of those surveyed described the organisational structures in place for ensuring user involvement in development, usually steering committees. This reflects both the historic difficulty many organisations have had in prioritising application development, and the difficulties faced by developers in achieving user participation in development. It also possibly reflects the perception that the 'how' of requirements definition is secondary to who should do it, which in turn suggests that the 'how' is not seen as a problem. Anybody can do it.

MANY DIFFERENT TYPES OF SYSTEMS STAFF DEFINE BUSINESS REQUIREMENTS

As one might expect, a wide range of different types of systems staff define the business requirements for application systems (Figure 2.1). In our survey, business analysts are marginally the most frequently mentioned group, closely followed by senior systems analysts and analysts. An interesting feature to emerge is that a number of companies have established a business group separate from the development group, and also some companies put the business analysts physically into the user departments.

Very few companies mention the use of analyst/programmers. Of course they could be referred to as systems analysts, but this suggests that extensive prototyping in requirements definition is not carried out in most of the participant organisations.

There are problems associated with both the use of a variety of systems staff, and also separate 'business' analysts:

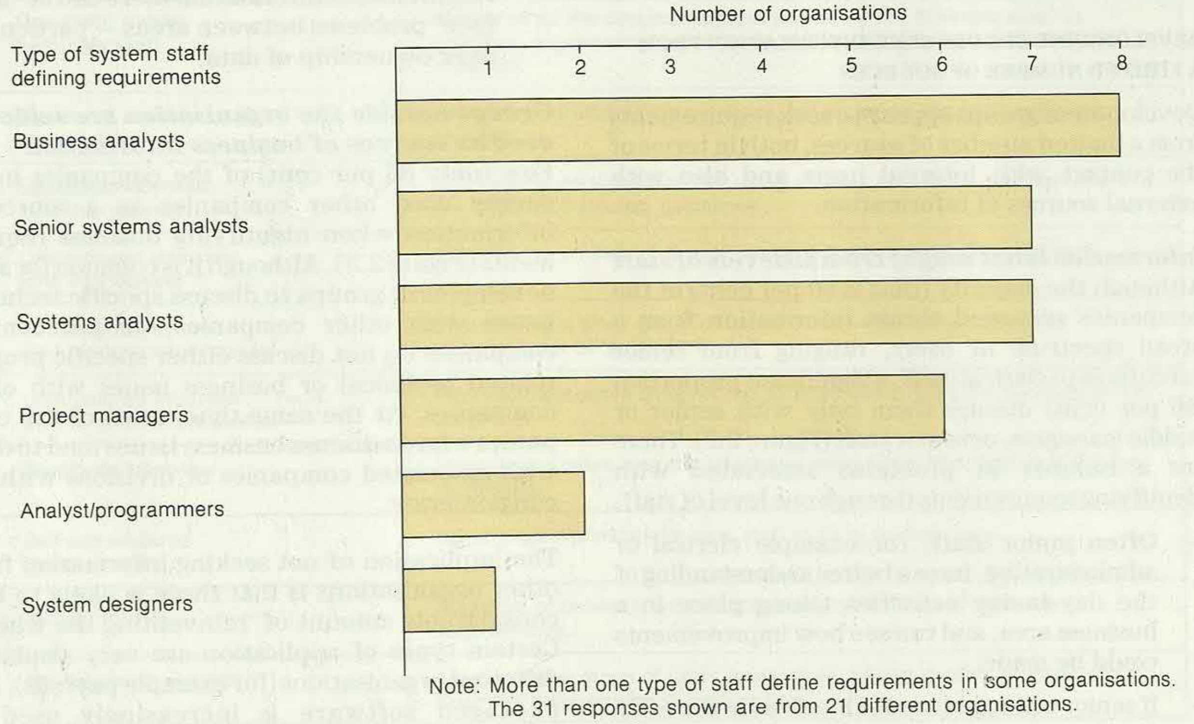
— It is possible that those companies with a large number of different types of developer involved in requirements specification do not see it as a separate function in its own right needing special skills and training.

— The companies using systems analysts and analyst/programmers to identify business requirements could be using staff who are not senior or experienced enough to delve behind the facts given by users in order to establish the real business needs.

— The introduction of a new organisational layer — the business group — can cause interface problems between developers and users. In some companies 'analysis paralysis' can set in, and developers watch awed as business analysts analyse the company or the specific application in ever more detail, over long periods of time, and no output ever reaches the developers. Also, some senior systems staff, when moved into business analysis have problems in 'letting go' their old role and tend to try to develop systems independently from the development group. Other business analysts with user backgrounds become highly enthusiastic 'techies' and are found struggling with the intricacies of programming rather than analysing business needs.

Other problems can arise in communication or 'redefinition' of problems. Communication problems are obvious:

Figure 2.1 Types of systems staff involved in defining business requirements



Note: More than one type of staff define requirements in some organisations. The 31 responses shown are from 21 different organisations.

> "You know the game where eight or nine
> People whisper down the line,
> And 'reinforcements to advance'
> Is '24 cents for the dance'."

Because of the fear of misunderstanding requirements, there appears to be a certain amount of 'redefinition' by systems analysts when they receive requirements specifications from the business analysts. Some is inevitable but extensive reworking can be prevented if structured techniques are used when defining requirements.

The involvement of users is clearly critical to the definition of requirements. In PEP Paper 1: 'Managing User Involvement in Systems Development', we explain our views on how users should participate in requirements definition in detail.

FEW DEVELOPMENT GROUPS PREPARE FORMAL RESEARCH PLANS

When asked how they decided which of the research techniques to use when defining requirements, most of the participants in the survey responded that this tended to be done by the specific analyst/project manager on the project, based on personal experience. There was no process of presenting a formal research plan. Without such a plan we believe that the requirements produced are likely to be insufficient and innaccurate.

The idea that system development is still a 'craft' industry not a 'science' is seen at its strongest in this phase of development.

DEVELOPMENT GROUPS SEEK INFORMATION FROM A LIMITED NUMBER OF SOURCES

Development groups appear to seek requirements from a limited number of sources, both in terms of the contact with internal users and also with external sources of information.

Information is not sought from all levels of staff

Although the majority (that is 60 per cent) of the companies surveyed obtain information from a broad spectrum of users, ranging from senior executives to clerical staff, a significant proportion (40 per cent) discuss them only with senior or middle managers, or senior staff (Figure 2.2). There are a number of problems associated with identifying requirements through one level of staff:

— Often junior staff, for example clerical or administrative, have a better understanding of the day-to-day activities taking place in a business area, and can see how improvements could be made.

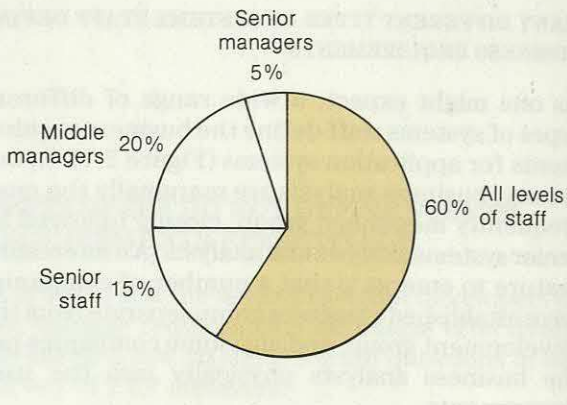
— If senior managers are not involved in the early stages of requirements definition, key elements of the business strategy may be missed, and systems could be obsolete or ineffective even before they are built.

— If middle managers specify requirements and senior managers merely 'sign-off' the cost of development, not only can the strategic element (above) be missed, but middle managers may provide requirements which are more extensive than is necessary resulting in over-engineered systems.

— Middle managers are responsible for specific areas of the business and if they provide the requirements information there can be 'interface' problems between areas — particularly over ownership of data.

Figure 2.2 Types of user involved in defining system requirements

Only 60 per cent of the organisations surveyed involve all levels of user staff in defining system requirements



Proportion of organisations involving only those users shown (%)

Groups outside the organisation are seldom used as sources of business information

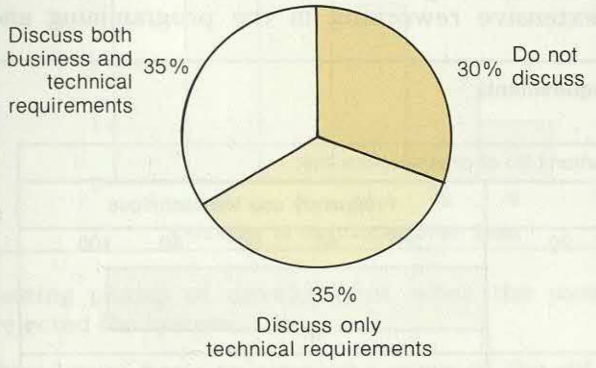
Few (only 35 per cent) of the companies in the survey used other companies as a source of information when identifying business requirements (Figure 2.3). Although it is common for some development groups to discuss specific technical issues with other companies, 30 per cent of companies do not discuss either specific project-related technical or business issues with other companies. At the same time, most of the companies who do discuss business issues tend to do so with associated companies or divisions within a conglomerate.

The implication of not seeking information from other organisations is that there is likely to be a considerable amount of 'reinventing the wheel'. Certain types of application are very similar in different organisations (for example payrolls), and packaged software is increasingly used in implementing such applications. But even where the applications are less similar, there are business lessons to be learnt from other organisations.

Apart from the above, there will be a need to share information with other organisations when developing systems in which commercial partners exchange information (for example, sales order and invoice information). These electronic data interchange systems will need to be carefully defined if they are to be successful.

Figure 2.3 Discussions of business or technical requirements with other organisations

Thirty per cent of organisations do not discuss either business or technical requirements with other organisations



Proportion of organisations (%) who discuss requirements with other organisations

Development groups do not exchange business information because:

- Some developers are either sceptical of the value, or even do not consider doing so in the first place.
- Some business managers believe they are experts in the area and there is no need to go outside.
- There are few mechanisms in place for exchanging information in this way, and developers do not know how to go about it.
- The application being developed is of considerable business importance, and competitive advantage might be lost if developers talked to external companies.

We believe that the last of these is the only really valid reason. Otherwise it is always worthwhile seeking information from external sources.

MANY INFORMATION COLLECTION TECHNIQUES ARE NOT USED

We identified a number of information collection techniques which are appropriate for system developers to use (Figure 2.4). During the survey,

Figure 2.4 Requirements definition techniques

Technique	Description
Document review (with checklist)	Review of all the documents/papers covering the business area, its activities, organisation, procedures, and systems using a predeveloped checklist.
Document review (no checklist)	Review of documentation as above, but on an ad hoc basis without a checklist.
Structured interview (one interviewee)	Interviewing an individual user using either a structured questionnaire or a set of semi-structured 'guidelines'.
Unstructured interview (one interviewee)	Interviewing an individual user using no questionnaire or guidelines.
Structured interview (two to three interviewees)	Interviewing a small group of two to three users using structured questionnaire or 'guidelines'.
Unstructured interview (two to three interviewees)	Interviewing a small group of two to three users using no questionnaire or guidelines.
Telephone interview	Interviewing users over the telephone using structured questionnaires or guidelines.
Self administered questionnaire	Issuing questionnaires for user staff to fill in on their own.
Group meeting	Meetings of between five to eight user staff to discuss business requirements.
Brainstorming session	Meetings of a group of users to generate a large number of ideas about business requirements in a short time.

PEP members were asked to state which techniques they actually used in requirements definition. After some initial confusion about information collection techniques and, in many cases, after prompting, the respondents indicated which of these techniques were used and which were not (Figure 2.5).

Most of the information collection techniques are seldom used by those surveyed, particularly telephone interviewing (except as a means of checking specific information), reviewing existing documentation, and structured interviewing. Indeed it is clear that unstructured interviews, and to a certain extent group meetings and document reviews without a checklist, are the most significant ways of obtaining information. Also, when the number of techniques used 'often' or 'very often' is analysed in more detail, we find that most organisations tend to use only a limited number of techniques (Figure 2.6).

Using inappropriate techniques can lead to statements of requirements which are incomplete, innaccurate, and even in some cases, misleading. During the survey, we were told of one development which went considerably over budget both in terms of cost and time. The reason for this was a statement of requirements which was produced by an inexperienced analyst who took the user manager's statements at face value in an unstructured interview and did not check the information given. As a result of this there was extensive reworking in the programming and

Figure 2.5 Techniques actually used in defining systems requirements

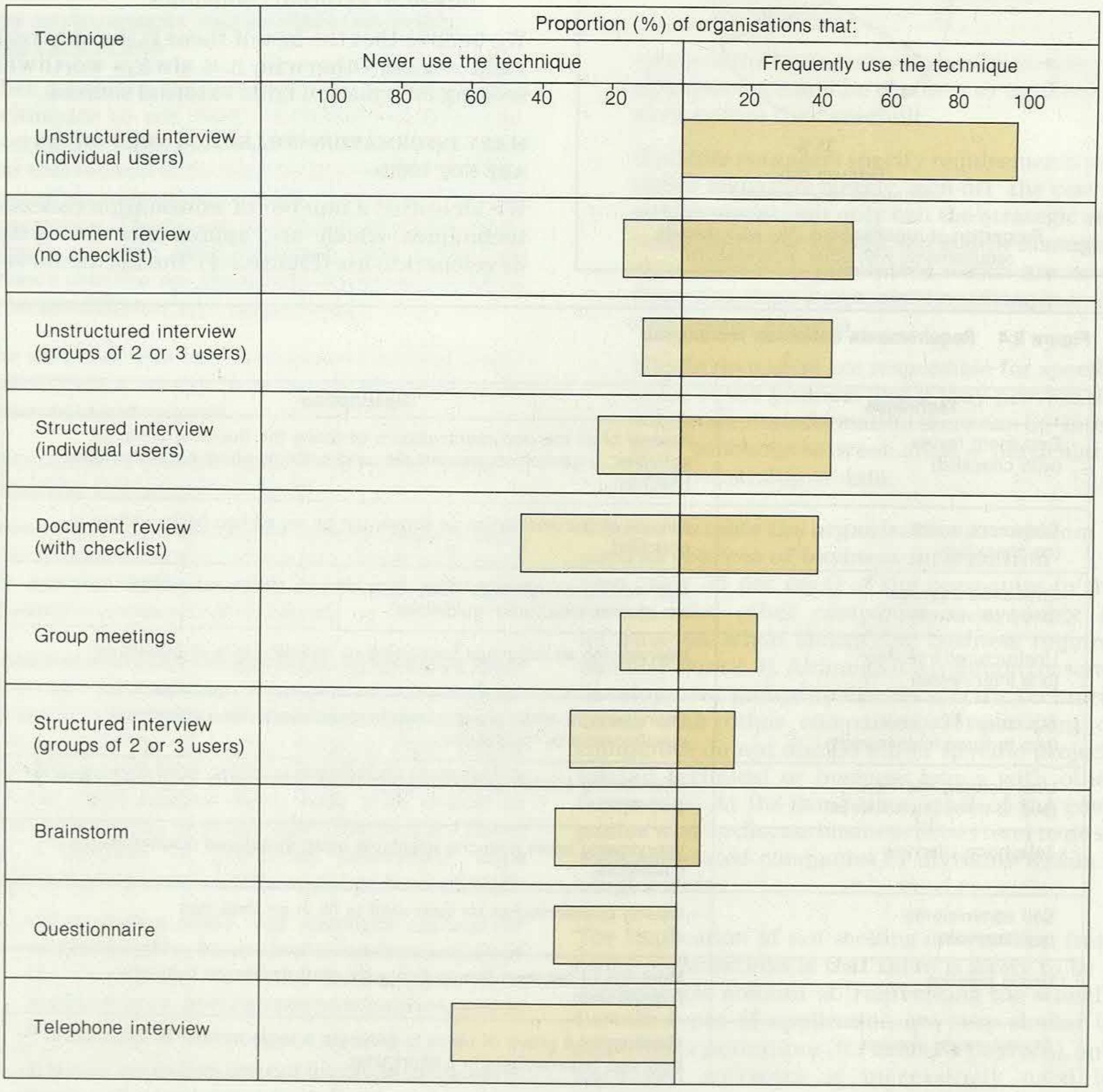
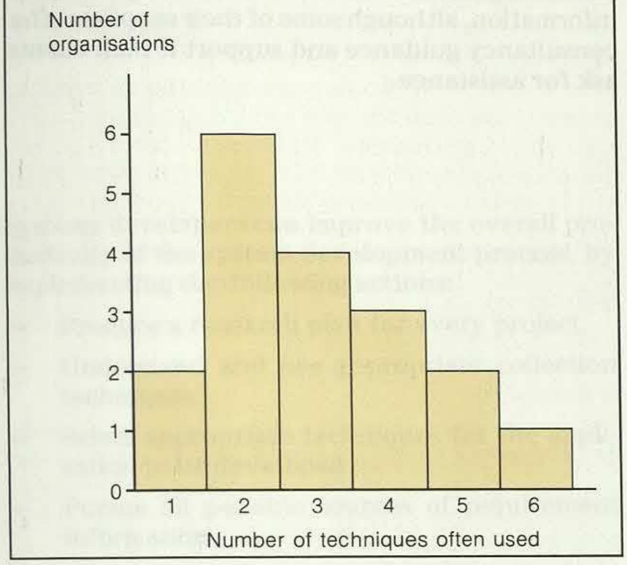


Figure 2.6 Use of multiple techniques in defining systems requirements

Most organisations typically use from two to four different techniques in defining systems requirements



testing phases of development when the user rejected the system.

Developers hope to overcome some of the difficulties associated with the techniques used, particularly unstructured interviewing, by the use of the structured analysis techniques since it is possible to document functional requirements and then check them with the user in an 'iterative' process. However this provides documentation for function and data only. Other key elements (such as who uses the system, the user-to-system interface, performance considerations, security, reliability, transferability, and so on), all of which lead to quality in systems, will be missing.

THE USE OF PROTOTYPING IS LIMITED

Although the vast majority of the companies contacted use prototyping in development (only two of them do not use it at all), they use it mainly for screen painting; very few use it to develop prototype systems, and even fewer use it to develop fully operational systems. The reasons for the delay in fully using prototyping are many and various, and are mainly related to the problems developers have had with this approach:

- It is difficult to adopt an 'iterative' approach to development when using sequentially orientated development approaches. The proprietary methods are only now suggesting guidelines for the techniques associated with using prototyping. Often, companies have had to develop their own in-house standards.
- Many companies have had difficulties with prototyping. For example:
  - Difficulty in persuading users to sign off the prototype—even though it is being actively used in production and has gone through a number of iterations.
  - Developers who see prototyping as a 'technical' solution, and who refuse to allow users any involvement in the process. One group apparently spent a short time (weeks) with users to identify requirements and then started to 'prototype' the system (in a locked room). Various initiatives by users to become involved were rejected, and ultimately at the end of a year's development the disgruntled users were delivered a system which did not meet their needs.
  - Hesitation on the part of some users to become involved in the process at all, and consequent delays in development.
  - Users who believe the prototype is the fully operational system and cannot understand why it is being 'taken away from them'.
- Prototyping is most easily used in small systems or where there is only one user. In larger systems or where there are many users drawn from different parts of the organisation, iterative development can be difficult to undertake in the absence of clear guidelines and standards.
- The fourth-generation languages themselves carry penalties in terms of reduced machine performance, particularly during development.

NOT ALL ORGANISATIONS USE QUALITY ASSURANCE TECHNIQUES IN REQUIREMENTS DEFINITION

Several organisations, but not all, pointed out that quality assurance techniques, such as walkthroughs, can be used in the requirements definition phase of development.

Walkthroughs with the users allow the requirements to be validated and enhanced. They are also useful in obtaining greater user cooperation at the start of the development process.

PROPRIETARY METHODS OFFER LITTLE GUIDANCE IN REQUIREMENTS DEFINITION

A review of the proprietary methods conducted by Butler Cox in 1986 showed that the proprietary methods offered little guidance in the area of requirements definition. They fall into three major categories when dealing with this phase of development:

- Project management methods such as Arthur Andersen's Method 1 and Hoskyns' Prism which provide advice on the sort of information to be collected (for example organisation charts, records of how current systems operate, and so on).

— Analysis/design methods, such as Yourdon, which 'assume' that the requirements information is already available.

— 'Integrated' methods, such as Information Engineering, which are based on the decomposition of requirements information and 'assume' the existence of some form of strategic overview and analysis.

None of the proprietary methods appear to offer practical guidance on how to collect requirements information, although some of their suppliers offer consultancy guidance and support if their clients ask for assistance.

Chapter 3

Improve requirements definition to improve productivity

Systems developers can improve the overall productivity of the system development process, by implementing the following actions:

- Produce a research plan for every project.
- Understand and use appropriate collection techniques.
- Select appropriate techniques for the application to be developed.
- Pursue all possible sources of requirement information.
- Use what tools are available, particularly fourth-generation languages.
- Quality assure the process.

PREPARE A RESEARCH PLAN FOR EVERY PROJECT

Preparing a research plan for every project is key to ensuring the successful definition of requirements. Preparing a research plan is a distinct activity in itself which takes time. Clearly this plan is a component of the overall project plan but specifically addresses the requirements definition phase. It should provide answers to the following questions:

- What are the objectives in this phase?
- Who will define the requirements?
- Who will provide the requirement information?
- How will the information be analysed?
- How detailed should the information requirements be?
- How will the requirements be presented?
- Who will the requirements be presented to?
- Who will validate and agree the requirements definition?
- What techniques will be used to collect the information?
- What are the constraints on collecting information in terms of time, cost, and the likely availability of key information providers and reviewers?
- What are the specific tasks to be carried out?

Once these questions have been answered, the first task in the project must be to undertake some preliminary analysis to determine the information to be gathered, and to structure lists of questions for interviews and for meetings. As a rule of thumb, the planning and preliminary analysis should take about one-third of the effort, collecting information should take the second third, and analysing and presenting the information the remaining third of the effort. Allocation of effort is independent of the size of the project. Planning should be done even in the smallest development project. (System rewrites are, of course, treated differently to new developments or system extensions.)

UNDERSTAND AND USE APPROPRIATE INFORMATION COLLECTION TECHNIQUES

Although each of the techniques discussed in Chapter 2 may be familiar to system developers, it is worthwhile ensuring that they are formally trained in each of them. Indeed some of the companies contacted during the research for this paper mentioned that developers were routinely sent on training courses in certain techniques, particularly interviewing. At best, this training will provide inexperienced analysts with new ways in which to obtain information. At worst, it will remind experienced staff of some of the basics which they may have forgotten. Figure 3.1 shows some of the 'do's' and 'don'ts' useful in conducting an interview. Even experienced interviewers sometimes forget some of these.

It is worth remembering that some of these techniques require a high level of personal skill and that although much of this is gained through training and experience, there is a certain amount which depends on natural ability. A particular example of this is the role of the facilitator in group meetings. Some companies who use facilitators to run formal meetings to discuss requirements have found that certain members of staff are better in this role than others. Relative skills in different information collection techniques are something which should be considered when assigning staff to projects. The information collection techniques should be chosen to match the needs of the application, not, as is so often the case, the skills and preference of the system developer.

SELECT APPROPRIATE TECHNIQUES FOR THE APPLICATION BEING DEVELOPED

It is vital to select the appropriate mix of techniques for the particular application which is being developed.

As a general rule, structured techniques are preferable to unstructured ones. Many organisations have already recognised this in the later stages of system development, and use structured analysis, design, and programming techniques. However, it is very clear from our brief survey that this is not so in the requirements definition phase. The most commonly used technique is unstructured interviewing. Clearly, the first step in selecting tools or techniques to aid information collection is to use structured techniques wherever possible — in interviewing individuals, or groups, and by the use of proforma questionnaires and checklists.

But there is then still scope to deploy a variety of techniques on any one project. In each application development project, there are a few key factors which can govern the information collection techniques that should be used. These factors include:

- The number of users to be contacted.
- How widely dispersed the users are.
- The number of different business areas, with different managers, the application will cover.
- The level of certainty/uncertainty about business requirements amongst the users.

Figure 3.1 Some 'do's' and 'don't's' of conducting an interview

Do:

- Create a friendly, informal atmosphere
- Explain why you are there
- Ask permission to take notes
- Know the questions you want answered (in advance)
- Ask questions which you know they can easily answer at the beginning of the interview
- Ask one question at a time and make sure it is understood
- Let the user answer the question and do not interrupt
- Stop rambling answers — but make sure they are not important first
- Watch 'body language' — both yours and theirs
- Check accuracy of statements
- Go over major points at the end
- Thank the user

Do not:

- Be arrogant or subservient; show respect
- Register emotion, for example surprise, shock, disagreement or agreement. ("You do what?")
- Patronise or lecture about technology
- Indicate your own views
- Break an appointment or arrive late
- Reveal information about the application under development inappropriately

But, in practice, there are also usually constraints on both the time and the budget available.

Figure 3.2 shows how each of the above factors relate to the techniques described in Chapter 2.

This figure can be used as an initial guide to select appropriate techniques for information collection. Two examples of how an application can be assessed are shown below:

Example 1:

A sales order application is needed for one division of an organisation with sales outlets spread over the country. There are slight geographical differences in how orders are taken. The requirements for the new system seem to be reasonably stable, being based on a system already in place. However the systems department has been given a very short time in which to develop the system.

Thus:

Number of users	—	high.
Geographic dispersion	—	wide.
Number of business areas	—	low (only one division).
Level of certainty	—	high.
Time available	—	short.
Budget available	—	not a constraint.

The most appropriate techniques are telephone interviews and/or self-administered questionnaires with perhaps a small number of face-to-face interviews with senior management.

Example 2:

The Board has requested a management information system to keep them in daily touch with the company's order book, sales, and revenue. They are all located in one building. Profits are high and this system is seen as a public relations exercise as well as a useful tool. However, they are not quite sure what they want in the system.

Thus:

Number of users	—	low.
Geographic dispersion	—	narrow.
Number of business areas	—	high.
Level of certainty	—	low.
Time available	—	not a constraint.
Budget available	—	not a constraint.

The most appropriate techniques are brainstorming sessions, group meetings followed by face-to-face interviews.

There are two features which emerge from the examples. Firstly, no one technique should be used alone; usually a combination of two or more techniques will help developers more fully define requirements. Secondly, we can only achieve a broad indication of which technique should be used using this assessment table. There are other factors also important in choosing an information collection technique. These include:

- Level of 'sensitivity' associated with the system. For example, will redundancies result from its introduction?
- Size and business importance of the system.
- Level of systems awareness and knowledge amongst users.
- Level of business awareness and knowledge amongst developers.
- Level of skill in defining requirements amongst developers.
- Culture of the organisation (for example, some organisations are familiar and comfortable with working in groups).

Note, however, that technical criteria are not relevant to the choice of information collection technique. However, there are technical considerations when prototyping is used as a technique in defining requirements.

Prototyping or iterative development often uses a fourth-generation language to build a system quickly, usually in conjunction with the user or users of the system. Prototyping can be deployed in three ways:

- To define and agree screen designs ('screen painting').
- To develop a 'prototype' system with limited functionality and associated screens.
- To develop a functional system, following several iterations where the 'prototype' is gradually modified to become the operational system.

The benefits of prototyping lie in two main areas. The first is that the fourth-generation languages used themselves can offer significant productivity benefits over earlier languages like Cobol. The second, and perhaps most significant from the requirements definition viewpoint, is in the increased ability to communicate quickly with the users, and to model their stated requirements in a lifelike system. These fourth-generation languages can thus be seen as a communications tool between developer and user, allowing them to 'speak the same language'.

PURSUE ALL RELEVANT SOURCES OF REQUIREMENTS INFORMATION

All relevant sources of requirements information should be pursued. Potential sources will be found both inside and outside the company.

Figure 3.2 Indicators for the use of different information collection techniques

Development project features		Information collection techniques					
		Document reviews	Structured interviews	Telephone interviews	Self-administered questionnaires	Group meetings	Brainstorming sessions
Number of users	High			✓	✓		✓
	Low		✓				
Geographic spread	Wide			✓	✓	✓	✓
	Narrow		✓				
Number of business areas	High	✓		✓	✓	✓	✓
	Low		✓				
Certainty of users	High	✓			✓		
	Low					✓	✓
Constraint on time and/or budget	Constraint	✓		✓	✓	✓	✓
	No constraint		✓				

Key: ✓ Indicates that the technique is particularly appropriate for a project with that feature

Inside the company, records of how activities are carried out are a useful starting point, particularly if computer systems are currently being used. The managers in charge of the area are obvious sources. The eventual users of the system can also provide valuable input and often it is worthwhile talking to colleagues in the systems group. Often the same or similar systems have been developed in the past.

Outside the company, packaged software suppliers and other companies who have developed the same or similar systems can provide valuable information. Even if packaged software is not ultimately bought, a knowledge of the business requirements leading to the development of the package can be useful.

Other companies can often provide very valuable requirements information. While it is difficult to contact organisations who are direct competitors, it is often possible to find companies in different industries who face similar issues. Examples include the process control industry, where food manufacturers can share information with oil companies, and warehousing, where chemical manufacturers can exchange information with car parts' manufacturers. Programmes such as PEP can facilitate this process by putting development managers from different companies in touch with one another.

USE WHAT TOOLS ARE AVAILABLE

There are only a limited range of tools available for assisting in system requirements definition. These include fourth-generation languages (discussed earlier in this report), office automation tools, analyst designer workbenches, questionnaires and checklists, and some documentation and communication tools. More specific tools, such as CORE (Controlled Requirements Expression from Systems Designers plc) can be useful but are typically limited to real-time applications.

OFFICE AUTOMATION TOOLS

A wide range of office automation tools can be useful in requirements definition, including word processing, standard formats and text templates (libraries of standard text), and graphics packages. Facsimile, electronic mail, messaging, and telex can also assist in collecting data from remote users and checking it.

The more people who have access to these tools, the more useful they are. Thus, if they are only available to the development group, not the user community, their real usefulness is lessened.

ANALYST/DESIGNER WORKBENCHES

Analyst/designer workbenches provide a means for the analyst to model the functional or data requirements in discussion with the user and change them immediately if necessary. They are particularly useful if the company has a top-down analysis of corporate or divisional requirements already available.

The key consideration to bear in mind when using these sort of tools is that they should not be used as an end in themselves, but as a means to the end of determining requirements.

QUESTIONNAIRES AND CHECKLISTS

Questionnaires and checklists are useful tools to use when defining requirements. Checklists ensure that nothing is missed. Questionnaires ensure both completeness and consistency of response. Self-administered questionnaires also enable information to be sought from a much larger sample of users than can be afforded when using interviews with individuals.

Checklists are relatively simple to generate, and standard lists can be adapted for specific projects.

Questionnaire design however is a skilled task, and poorly designed questionnaires can not only provide misleading information when analysed, but can also give users a poor image of the systems development group.

RECORDING AND COMMUNICATION TOOLS

Recording tools include tape recorders (both audio and video). They can be useful as a means of reliably recording information in face-to-face or telephone interviews and group meetings. However it is vital to have the permission of the interviewee or group members in advance. Also, expect that the meetings will be somewhat stilted at first. The main drawback of using these recording tools is the amount of time it takes to play them back when the information they contain is being analysed.

Communication tools obviously include the telephone as well as the various forms of electronic messaging mentioned above. Where the application warrants the cost, videoconferencing or audioconferencing may also be used for group meetings. These are not so effective as face-to-face meetings, but can have distinct advantages in terms of time and cost. They will only normally be used if the company already has such facilities available.

QUALITY ASSURE THE PROCESS

As many of the participants in our survey suggested, using quality assurance in the requirements definition process is essential and will help ensure quality and promote productivity.

It is vital to include the user in this process. For example, they can attend walkthroughs, join in peer group reviews, or review the requirements documentation being produced by the analyst or analysts.

As a minimum, the initial research plan and the final documentation defining the requirements should be reviewed together with progress at at least one interim review during the definition phase. This will prove of value on even the smallest project.

BUTLER COX

P,E,P

Butler Cox

Butler Cox is an independent international consulting group specialising in the application of information technology within commerce, industry and government.

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.

The services provided include:

Consulting for Users
Guiding and giving practical support to organisations trying to exploit technology effectively and sensibly.

Consulting for Suppliers
Guiding suppliers towards market opportunities and their exploitation.

The Butler Cox Foundation
Keeping major organisations abreast of developments and their implications.

Multiclient Studies
Surveying markets, their driving forces and potential future.

Public Reports
Analysing trends and experience in specific areas of widespread concern.

PEP

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

It provides practical help to system 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 position papers and forum meetings common to all subscribers.

Productivity Assessment
Each subscribing organisation receives a confidential management assessment of its system 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.

Position Papers
Four PEP position papers are produced each year. They focus on specific aspects of system development productivity and offer practical advice based on recent research and experience.

Forum Meetings
Each quarterly PEP forum meeting focuses on the issues highlighted in the previous PEP paper, and permits deep consideration of the topic. They enable participants to exchange experience and views with managers from other subscriber organisations.

Topics in 1987

Each year PEP will focus on four topics directly relating to improving systems development and productivity. The topics will be selected to reflect the concerns of the subscribers while maintaining a balance between management and technical issues.

The topics covered in 1987 are:

- Managing user involvement in systems development.
- Using tools to improve productivity.
- Planning and managing projects effectively.
- Using methods to improve productivity.

BUTLER COX

Butler Cox & Partners Limited
Butler Cox House, 12 Bloomsbury Square,
London WC1A 2LL, England
☎ (01) 831 0101, Telex 8813717 BUTCOX G
Fax (01) 831 6250

Belgium and the Netherlands
Butler Cox BV
Burg Hogguerstraat 791
1064 EB Amsterdam
☎ (020) 139955, Fax (020) 131157

France
Butler Cox SARL
Tour Akzo, 164 Rue Ambroise Croizat,
93204 St Denis-Cedex 1, France
☎ (161) 48.20.61.64, Fax (161) 48.20.72.58

Germany (FR)
Butler Cox (Deutschland) Ltd.
Richard-Wagner-Str. 13
8000 München 2
☎ (089) 5 23 40 01, Fax (089) 5 23 35 15

United States of America
Butler Cox Inc.
150 East 58th Street, New York, NY 10155, USA
☎ (212) 486 1760

Australia
Mr J Cooper
Consultants (Computer and Financial) plc Australia
Level 5, 303 Pitt Street, Sydney 2000, Australia
☎ (02) 2870400, Fax (02) 2870450

Ireland
SD Consulting
72 Merrion Square, Dublin 2, Ireland
☎ (01) 766088/762501, Telex 31077 EI,
Fax (01) 767945

Italy
SISDO
20123 Milano, Via Caradosso 7, Italy
☎ (02) 498 4651, Telex 350309, Fax (02) 481 8842

The Nordic Region
Statskonsult AB
Stora Varvsgatan 1, 21120 Malmo, Sweden
☎ (040) 1030 40, Telex 12754 SINTABS

Spain
Associated Management Consultants Spain SA
Rosalía de Castro, 84-2°D, 20835 Madrid, Spain
☎ (91) 723 0995