ALVEY PROGRAMME

SOFTWARE ENGINEERING STRATEGY

November 1983

A L V E Y   P R O G R A M M E

S O F T W A R E   E N G I N E E R I N G   S T R A T E G Y

N o v e m b e r   1 9 8 3

Mr D Talbot & Dr R W Witty

Alvey Directorate
Millbank Tower
Millbank
LONDON
SW1P 4QU

Tel: 01-211-0050

ALVEY PROGRAMME

PART I

SOFTWARE ENGINEERING STRATEGY

1. GOALS AND OBJECTIVES

1.1 The Central Goals

Future IT products can be expected to be more complex than those of today and thus to place greater demands upon the people building them. The IT industry must meet this challenge, even though there is a growing recognition that system development techniques are inadequate for the large systems of today, let alone those of tomorrow.

It is accepted today that the development of a substantial new computer system carries a number of significant risks and it is by no means uncommon for such systems to be delivered late, over-budget and incapable of meeting the complete requirements of the purchaser. Some systems, after considerable expenditure of human effort and money, fail to materialise at all.

Skilled programmers are a scarce resource which is not being used efficiently. The industry is fragmented by organisation, by language and by target computer. One result of the consequent lack of commonality of environment or concentration of resources is that many programmers are not provided with even the simplest programming aids, let alone sophisticated ones. The economies of scale necessary to justify their introduction have not been perceived to exist.

Despite these problems, the UK does not lag behind other countries in software engineering, except perhaps the USA. The UK is certainly regarded as the leader in Europe in this field. Efforts to improve software engineering practice are crucial if important developments in technology are not to be wasted or cast into disrepute through poor production methods. The UK must not allow other countries to overtake it, for if it does, UK research work will be exploited by other countries to the detriment of our industry.

Software engineering may be considered as having two major goals for the future:

- improved quality ie satisfying criteria such as performance, reliability, security, on-schedule delivery and meeting the needs of the user;

- improved productivity ie reducing cost, not just of the development but of the life-cycle as a whole, including maintenance and future evolution.

Current software practice is centred on the programming process, and depends strongly on the skills, experience and resources of individual workers. Significant problems frequently result from inadequate effort being devoted to the front end of a development, notably concept formation, requirements definition, and design. Although there have been some efforts to study these problems, as well as interesting advances in both design verification and code verification, relatively little work has been devoted to integrating all of the stages into a common framework useful in production environments. Significant improvements in software productivity will be achieved when the current practice of repreated 'reinvention of the wheel' is replaced by the widespread re-use of prefabricated components. In the future then, software practice will tend to focus more on methodology, design, and component reuse and less on individual programming skills.

System design must include not just software design, but also hardware considerations. A narrow view of software engineering as just a collection of techniques to produce efficient software is not adequate. Software engineering should be aimed at the development of high quality systems, ie reliable, secure, efficient and easy to use, in a way that integrates hardware and software-based design criteria. In the future it must become information system engineering, not just software engineering.

In the short term, the UK cannot afford grossly inefficient utilisation of its scarce skilled programming resource. Introduction of simple tools on a wide scale is an essential first step in increasing programmer productivity, and also in the educational process needed to prepare for the later exploitation of more sophisticated methodologies and tools.

1.2 Major Objective

To help achieve the general goals of improved Quality and Productivity the Software Engineering component of the Alvey Programme will be focussed towards a strategic goal - that in 1989 the UK should be a world leader in Information System Factories (ISF). This goal is highly ambitious and competitive, as are the goals of the Japanese 5th Generation Project. The ISF objective implies a series of sub goals both in technology and timescale. The Alvey Software Engineering component will be judged on its ability to show that UK industry has increased both its software development productivity and software product quality as a result of striving to achieve the ISF. The strategy given in this document outlines the route towards the ISF with planned interim spin-offs so that productivity and quality gains may be achieved prior to the emergence of the ISF.

What is meant by an Information System Factory? Today, the production of most application-specific hardware/software systems - such as a banking network, a corporate management information system or production control system - does not in general make great use of development tools. In that sense it is not capital intensive. The application-specific part of the Information Technology industry is characterised as a cottage industry. It is predicted that it will not remain so for long; that to stay competitive in producing large, reliable, application-specific systems, IT companies will have to make a large investment in some kind of production facility. Exactly the same criteria will apply to manufacturing software products. This expensive facility - part hardware, part software, part stored knowledge - is an Information System Factory.

1.3 Subgoals and Directions

Defining a concrete and ambitious strategic objective crystallises a number of more general but worthwhile aims as subgoals along the route to the main goals. Many questions about the balance and direction of the whole programme can be judged by their contribution to the main goals.

1.3.1 Use

An equally important part of the programme will be to create a climate in which advanced software engineering methods are in demand. This must start with an immediate programme of investment in and use of today's technology, with associated training, measurement and evaluation. This will need to be supported by education in 'formal methods' to prepare for the widespread introduction of specification and verification.

1.3.2 Measurement

The ISF will only succeed if it can bring radical improvements in software quality and productivity. These two concepts are notoriously difficult to pin down, and certainly it is not currently known how to measure them. The programme must face up to the difficult task of developing metrics for quality and productivity. Subgoals must be set for achievement. Performance must be reported against these goals. Finally it must be possible to measure the impact of an Information System Factory.

1.3.3 Distributed Working

Immediate use must be made of Wide and Local Area Networks to link co-operating designers and programmers tackling common development and production tasks. This requires investment in the use of current network technology. Measurements must be made of the improved performance flowing from these investments.

1.3.4 Research

Substantial research tasks must be undertaken starting now with the goal of incorporating successful outcomes into products from 1985 onwards. This requires co-operation between industry and universities. The correct balance must be struck between basic research, development, practical experimentation and importing other people's ideas. The goal of having a commercially viable Information System Factory by 1989 will provide a focus for research. It tips the balance towards practical experiments, development, and a readiness to exploit other people's ideas, rather than concentrating solely on basic research for "scheduled breakthroughs".

1.3.5 Short term exploitation

A number of subgoals must be established and met along the way to the selling and exporting of tools over the coming years. The establishment of strong sales organizations in the key markets is vital. Some companies have started already in software products. Many more must make this investment. The programme can make an extremely valuable contribution through support and direction of this investment. This will probably be most effectively achieved through close cooperation with other Government schemes which are more specifically aimed at product development and marketing, thereby constructing a smooth 'pipeline' from Alvey R & D through to product sales.

1.3.6 Standards

The programme must support the development of standards. These will vary from major international activities eg. ISO language standards, through to informal, Alvey specific, tools interfaces. Close cooperation must be established with other Government and Industry standards initiatives. It is anticipated that the increasing use of formal methods will improve the foundation and creation of standards.

1.4 What Will Happen in Any Case

The Alvey SE strategy is based on the prediction that the production of application-specific information systems will cease to be a cottage industry and become a capital-intensive industry.

The main reason has to do with software quality, in the widest sense of the word. Expectations of software quality, both within the industry and without, are very low. Programmers expect to have lots of bugs in their code, and the public expect computers to send them stupid invoices. This situation is not confined to the UK; it is worldwide. British standards of software quality are relatively high, while low in an absolute sense. This situation cannot last indefinitely. In the hardware field, one manufacturer (Tandem) has grown spectacularly by offering high reliability at a premium. This has been done against a background of hardware from IBM and others which is already highly reliable. The incentives to do the same in software, and the potential payoffs, must be much higher given the current poor quality of software. It seems highly likely that someone soon will "do a Tandem" in software, and either keep the method to himself or sell it very expensively. The Japanese are certainly trying, as are the Americans and the French. Without concerted action, the UK is bound to become an importer of this technology. If the UK is prevented from importing such technology then the industrial consequences could be very serious.

A number of other current trends are leading towards the 'capitalisation' of the software industry - the growing complexity of software systems, which demands new techniques and computer assistance to manage it, the dawning awareness of the importance of programming support environments, and the emergence of software packages which demand new skills to integrate them in particular applications. Finally, there is the emergence of non-Von Neumann architectures and VLSI, which are inevitably mixing the software and hardware design problems, making both more complex. All these are creating larger and more complex problems, which cannot be solved without a radically new level of automation and mechanical assistance.

2. THE CHANGING NATURE OF SYSTEM DEVELOPMENT

2.1 Summary

The expected changes that will result in the most significant increases in cost-effectiveness of software development over the next ten years are the following, listed in approximate order of expected impact.

In the short term

1. incremental changes in programmer productivity through the more widespread use of design methodologies and tools

2. the coming together of methodologies and tools for the entire development life-cycle within integrated project support environments (IPSEs)

3. growing standardisation of development methodologies as a consequence of 2.

4. further refinement of suitable high-level programming languages appropriate to the integrated development methodologies

5. growing interest in, and use of, formal specification methods and extension to animation

6. automatic software generation techniques in limited form, probably first in the area of commercial systems built around Data Dictionaries.

In the medium term

7. spread of powerful networked, personal workstations

8. consolidation of the use of formal specification methods coupled with verification and growth in use of (semi-) automatic software generation

9. development of reusable software and hardware modules, rigorously tested and formally documented

10. second generation IPSEs adapted to support activities 8 and 9 above, coupled with greater use of higher-level languages.

And in the longer term:

11. the consolidation of the developments above into Information System Factories, coupled with the use of Intelligent Knowledge Based Systems, to provide 'automatic' assisted system development from user requirements expressed in high-level terms appropriate to the application rather than the implementation.

The crucial, and inter-related, technical developments underlying those changes will be:

1. integrated system (software and hardware) development methodologies supported by programming tools, administrative procedures and management information in an integrated environment

2. formal specification, leading to 'animation' and verification

3 reusable software and hardware components

4 automatic software generation

5 measurement and quality assurance and certification

These are discussed more fully below.

2.2 Integrated System Development Process

One view of the system development life-cycle is the following:

REQUIREMENT SPECIFICATION OVERALL DESIGN DETAILED DESIGN CONSTRUCTION TESTING	typically costs 20-50% total development budget
OPERATION	- not usually quoted

RECTIFICATION & EVOLUTIONARY DEVELOPMENT	euphemistically called maintenance	typically costs 50-80%

Many design methodologies and software tools exist and are in sporadic use today, but the state of the art leaves much to be desired. For however good some of the tools may be, there are two serious problems.

First, they do not support a development methodology or capture any data relevant to the management of the development process. Second, most tools support coding activities but fail to support the life-cycle in its entirety, or even fail to be compatible with other relevant tools. There is a need for more tools to assist with software specification, design, testing, rectification and development, as well as with management of software projects; and there is a need to integrate them into a coherent life-cycle support environment built on a database.

Recently there has been widespread recognition of these problems, with a resulting effort to develop better tools; a prime example is the growing work on Ada Programming Support Environment (APSE), which should lead to a qualitative and quantitative improvement over today's state of the art. Viewed in the wider context of software engineering advances generally, two important short term benefits from such work should be increased programmer productivity in the technical tasks of project development and increased management awareness and control, leading to better decision making and costing. Moreover, the growth in use of integrated project support environments (IPSE) should provide the framework within which subsequent advances, such as improved specification and verification methods, can take place. This last point argues for a need for flexibility in IPSEs. They must not be closed systems incapable of accommodating improved techniques as these are developed elsewhere.

Whilst there are a number of issues still under debate, there does seem to be fairly widespread agreement on certain key characteristics that these environments will display.

First, and of crucial importance, there will be far less emphasis on the actual source text of the program than there is at present. Typical current practice focuses far too much attention on the source code representation of a program and far too little on other representations - expressions of requirements and various levels of specification. The tools which are most commonly employed are those concerned with manipulating and testing the source code representation. Yet most software projects that are 'unsuccessful' by some measure have already gone irretrievably wrong by the time that the first line of source code has been written. If there is to be real progress on the issues of effectiveness and cost then attention must be shifted from the code to requirements and design, and projects must be far more concerned with the 'higher level' representations. (Note that such a shift of attention is entirely compatible with an aproach which emphasises re-use of existing components rather than always developing everything from scratch.)

Second, the environments will support a high degree of project visibility and traceability. At any stage of a project all relevant information will be readily available and there will be a proper basis for measurement of progress and detection of problems. For any identifiable activity there will be a record, not only of the end product of that activity, but also of the decisions (both positive and negative) which were taken during that activity.

Third, the environment will support various kinds of control. Management control, access control and configuration control all play an important part in addressing software effectiveness and software costs.

Reviewing the three issues above - emphasis on 'higher level' representations, visibility and control - leads to an inevitable conclusion: any given project employing such an environment must follow a defined methodology. This is not to say that the environment offers only a single methodology, but it is necessary for any given project to employ some defined methodology, and it is necessary for the supporting environment to 'recognise' this methodology (or at least certain aspects of it). Really, it is the methodology which addresses the issues of software quality and cost. The degree to which these issues are addressed depends upon the quality of the methodology and how well it is supported.

2.3 Formal Specification

The first qualitative change that will occur in system development will be the use of formal specification techniques. It is a large leap from today's practice to automatic program generation on a large scale, to proving theoretically that systems meet their requirments, to easy re-use of system components; but in each case the first step is formal specification.

Today's functional specifications are written in English, often with a liberal sprinkling of design detail in the difficult parts.

Specifications written in natural language have the major defects that:

a. they are imprecise, ie are subject to conflicting interpretations

b. they may be logically inconsistent without the fact being apparent

c. they are apt to be incomplete

d. they cannot be used for (mechanically assisted) formal reasoning.

Use of natural language does not force the specifier to be precise at all times. In some cases he may be unaware of imprecision, which thus slips through; in other, he may decide to gain precision and by default the method chosen will probably be to take some design decisions and specify the requirement in terms of an implementation. Neither result is satisfactory.

The development of formal specification techniques should ultimately overcome these difficulties and lead to complete, precise specifications which do not contain any unnecessary design detail. Experience has already shown that efforts to translate natural langauge specifications into a logical form show up inconsistencies, ambiguities and omissions.

During the development of the complete specification, particular specifications can be 'animated' in the sense that their logical consequences can be explored. Questions such as What will happen if..? can be answered precisely, and the specification improved or modified as appropriate. In this way, purchaser and supplier can gain the clearest understanding of the system requirements. Another use would be simulation of critical aspects of the system, for example the user interface of a Command and Control System, so as to give the customer an early understanding of them.

Ultimately, formal methods can provide a very clear contractual basis for the statement of requirements and thus help to avoid disputes about whether the system meets the requirements or not.

Numbers are hard to come by, but it is probably fair to say that most computer systems have to change after only a limited period of operation because the true operational requirements are, with the benefit of hindsight, perceived to be different from those originally requested. It has been argued that that is not so much a problem as an inherent characteristic of the real world which must be catered for in the development process. Systems must be designed to be capable of evolution. A rigorous path from specification through to implementation, with all the steps recorded, is essential if newly understood requirements can be fed in again at the beginning of the process without requiring a complete rewrite of the system.

2.4 Re-usable System Components

Today, hardware is thought of in terms of components whose behaviour is well understood and which can be put together in a number of ways to build a system. In the future, software will more and more come in packages until it too can be regarded as providing a set of component parts out of which software or mixed hardware/software systems can be constructed.

A number of trends will work together to bring this about. First, packaged software will account for a higher proportion of software sales to meet the enormous need for inexpensive software for personal, home or other small computer systems. Tailor-made software will be too expensive for this market and suppliers competing to reduce production costs will find it necessary to use mass production techniques, eg standardised design techniques, specialised tools, integrated project development environments.

Second, skilled programmers are a scarce resource and will continue to be so. Development techniques which provide a path away from today's labour-intensive methods will permit levels of production control and documentation adequate to the development of truly re-usable software.

Third, design by components appears to offer the only solution to the problem already encountered today that some systems are so large and complex that their operation is hard to comprehend, their performance impossible to predict and their design impossible to optimise. Design in terms of components may permit only theoretical sub-optimisation but in practice this may be vastly superior to what could be obtained otherwise; and the ability to predict performance and cost, in advance of implementation, will be a major benefit.

2.5 Automatic Software Generation

Automatic software generation is in use now in a limited way, and is a very powerful technique for producing commercial software of the type that consists of simple, repetitive processing applied to a complex database. The development of data dictionaries in the commercial sphere, the trend to put the structure of applications into the database rather than into the programs, will encourage automatic software generation so that it can be expected to be a common technique in transaction processing within five years.

The experience thus gained, coupled with advances in specification techniques and the availability of a wide variety of software components, will subsequently enable automatic software generation to be applied to increasingly more complex processing tasks. It is here that Intelligent Knowledge Based Systems can be expected to make their greatest contribution to Software Engineering, in particular in determining and enforcing consistency of specifications and of the transitions from requirements specification to design and from design to implementation.

2.6 Measurement and Quality Assurance & Certification

Today it is difficult to predict the costs and timescale of a software development project, to measure the progress and productivity of the project team and to measure the quality of the finished product.

The widespread adoption of integrated project support environments built on database technology will facilitate new research on the quantitative aspects of software development. A significant improvement in management effectiveness, productivity and product quality will occur when respectable metrication is introduced into the software development process.

Quantitative assessment of the benefits of new tools and techniques will provide a major stimulus to the introduction of further new techniques and further research, as hard-headed senior managers will be more easily persuaded to make the necessary investment funds available when presented with reputable, quantitatively argued cases with measurable pay offs.

Metrics and formal methods are the keys to effective quality assurance. The developers of good quality assurance techniques will enjoy a significant commercial advantage. The current shaky reputation of software will mean that the ever broadening range of customers will gravitate towards products bearing something akin to the BSI kite mark for simple products and components. Customers wanting more sophisticated products will favour suppliers who can offer independent, top class quality assurance and certification as part of the legal contract.

3. STRATEGY

3.1 Summary

Consultation has shown that there is very strong agreement in industry, Government and the academic community on the technical directions that the software engineering programme should take.

The Alvey SE Programme has as its long term objective the creation of the 3rd generation IPSE or Information Systems Factory. This is predicated on technical progress in the two crucial areas of:

1. PRODUCTIVITY
2. QUALITY

To ensure continuous benefit during the period preceeding the achievement of the ISF the SE Programme proposes a strategy which encourages intermediate levels of technology transfer by encouraging not just research but:

i. Exploitation: efforts to ensure that existing methods are effectively used and their benefits gained by industry as a whole, and continuing efforts to bring the fruits of research out into industrial use, with the associated investment and training.

ii. Integration: development of integrated methodologies and sets of tools for hardware and software development covering all phases of the system life-cycle.

iii. Innovation: research and development to extend the methodologies and techniques of software engineering.

To give a feel for the activities which will be covered by innovation, integration and exploitation figure 1 shows the system development life cycle subdivided into

1. Methods and processes - how things are developed.

2. Management - monitoring and control of 1.

3. Environment - the workplace, tools and equipment with indications of where in the classification various key elements of the strategy occur. Figure 1 is a summary which is expanded in the following sections.

STRATEGY	**Innovation and Understanding**	**Integration and Implementation**	**Exploitation and Evaluation**
Methods and Processes	Specification V & V Reliability Quality Metrics Reusability	Blend techniques into life cycle method for both hardware and software	Measure use of IPSE
Management	Models of development and mainte- nance processes and methods	Integrate development methods with management techniques	Evaluate use of IPSE
Environment	Influence on Productivity and Quality MMI, IKBS, DCS	Build IPSEs	Make IPSE available via Centres

Figure 1

3.2 Exploitation

Today, most small to medium projects in the UK (and elsewhere) confine their use of tools to simple text editors, compilers or assemblers, linkers and simple debugging aids. Occasionally, a design technique such as Jackson Structured Programming or MASCOT will be used.

In the rare cases that a more systematic approach is felt to be necessary, and additional tools to support the approach are required, they are commonly developed ad hoc, on a project specific basis, and thrown away when the project is complete. The high cost of this approach both in terms of tool development and in re-training staff to use new tools and techniques for each project, has militated against the systematic use of tools throughout the UK software industry.

There is therefore great scope for improvements in software quality and productivity, in the short term, by encouraging the widespread acceptance and use of even simple tools, of the kinds currently in use in more restricted environments. Three things are needed to bring these improvements about:

i. provision of a set of tools in a standard, compatible form

ii. measurement of the effect (positive or negative) on quality and productivity due to the new methods, tools and training.

iii. education of both management and software staff; management must be shown that investment in tools does pay off, and software staff must be educated in the systematic development and production methods that enable the tools to be used cost-effectively.

A short term attack on these three factors will be crucial not only in bringing about the much-needed improvements in quality and productivity, but also in providing wide appreciation of the nature and benefits of software engineering and the demand for more advanced techniques.

In the longer term, continued efforts will be needed to ensure that the results of research are developed and exploited. The 'development gap' between research and production has been a problem in Britain for many years, and the software engineering programme will tackle it directly by commissioning innovative research and development projects rather than just funding research. Moreover, the main objective of the proposed 'Software Production Centre' (see section 3.6) is to make advanced tools directly available to British industry for experimental evaluation and genuine production work.

3.3 Integration

The second major need identified is for Integrated Project Support Environments (IPSE). The common understanding of an IPSE is that it should contain a compatible set of specification, design, programming, building and testing tools, supporting a development methodology that covers the entire life-cycle, together with management control tools and procedures, all using a central project database. That is already a very demanding requirment, exceeding that of the Ada APSE, but even then it does not go far enough. It does not cover multiple-language development; it does not cover mixed hardware and software development; it does not cover reusable components.

There is certainly no agreement that one particular programming language meets all foreseeable needs, though there are individual proponents of this view for different languages. There is also considerable investment in software in the languages of the 60s and 70s, which will tend to prolong their life for reasons of compatibility, cost of re-training and so on. This multiplicity of languages, coupled with a recognition of the need to move towards re-usable software, argues for multi-language IPSEs where systems can be built out of components in a variety of languages.

Similar considerations apply to mixed hardware and software systems. It is clear that there are enough similarities between the hardware and software design processes, and the administrative and management procedures appropriate to them, for there to be benefit in using one IPSE for hardware and software development. Furthermore, it is important that the requirements analysis, functional specification and much of the design work can be done independently of decisions whether particular modules should be implemented in hardware or software. For such modules, their function must be defined, their place in the overall design established and their performance requirements known; economic, timescale and other criteria may then be used to determine how they should be implemented.

A fully integrated IPSE as just described is exactly the Information System Factory that is the major objective of the programme. It is a long term objective, but it is important to be clear about what the objectives are in order to see how to move towards them, and in particular to determine the role of UNIX and Ada APSE developments in this process.

One conclusion that emerges strongly is that there is still a great deal of research and development to be done before such an integrated PSE can be built. Two important areas needing R & D are:

i. formal, rigorous methods of specification of requirements, and techniques to express designs and determine how far they meet their specifications for performance, reliability, correctness etc;

ii. methods of structuring software or hardware system components for wide re-use; the nature of their interfaces to each other, the appropriate types of global design to incorporate them; how to document them; how to search for and locate them.

The Alvey programme will include one or more evolving IPSEs, which not only bring together existing tools and procedures to improve development cost-effectiveness in the shorter term but are also capable of incorporating new techniques that emerge from relevant R & D projects.

3.3.1 ISF and the Three IPSE Generations

The Integrated Project Support Environment (IPSE) is a major product objective of the programme and a crucial mechanism for blending together the results of individual research projects. The blending process is itself a research topic. The blending might well prove to be more important that any of its constituents when judged in terms of commercial success. The programme will proceed as follows.

(1) Commission development and creation of three generations of IPSE:

```
1st)                ) file
2nd) generation IPSE ) database
3rd)                ) knowledge base
```

(2) Versions of each generation of IPSE to be sited in SPC (section 3.5) and NQCC (section 3.6) and selected organisations where IPSE impact on quality and productivity can be monitored and reported.

(3) Cooperate with and incorporate aspects of other Alvey areas towards ISF eg CAD for VLSI, high resolution displays, expert systems for programmers.

3.3.2 The 1st Generation IPSE

UNIX will be used as the basis for:

(1) The 'Exploitation' Tools Propagation exercises.

(2) The 1st generation (file based) IPSE.

UNIX is rapidly becoming a de facto standard over a very wide range of systems and organisations and therefore offers the prospect that:

- There will be many developments for UNIX which can be taken advantage of by the Alvey programme.

- The market for UNIX-based development environments and tools is large and growing.

These factors should minimise the amount of tool integration and development needed to improve today's UNIX environment into a genuine 1st generation IPSE.

This is not to say that the Alvey programme is endorsing UNIX as a standard; UNIX will be used as a starting place. Nonetheless, it is envisaged that an active UNIX community will come into being in the early years of the programme, supported by communications network facilities.

3.3.3 The 2nd Generation IPSE

The second generation IPSE contains two major components not found in the 1st generation IPSE:

(1) Database-based tool set (rather than file-based) eg CADES.

(2) Support for geographically distributed project teams e.g. Newcastle Connection.

As (1) and (2) above are somewhat orthogonal it is expected that several approaches will be attempted, including the evolutionary development of the 1st generation, UNIX-based IPSE as well as the 'clean sheet', non UNIX attack, possibly via intermediate steps which will contain one, but not both, of the distribution and database components.

The 2nd generation IPSE software will run on new hardware; developments in cheaper CPU power, cheaper, high resolution colour graphics, and non keyboard input-output devices, for instance, will facilitate productivity gains due to improved man-machine interaction. The 2nd (& 3rd) generation IPSE will require new hardware based components such as:

1. Single user workstation costing £5K with A3 black and white 2K x 2K pixel graphics.

2. Colour single user workstation costing £10-20K with

    - 2K x 2K pixel A3 screen
    - 10 MIPS power CPU
    - 32K microcode store
    - 10 Mbytes physical memory
    - 32 bit arithmetic and data paths
    - 32 bit virtual address space per process
    - hardware cache, paging, floating point
    - hardware graphics support
    - sophisticated i/o devices.

3. 100 Mbits/sec local area network.

4. Gateway to high speed (greater than 1 Mbit/sec) wide area communications.

5. LAN servers for files and databases.

6. High quality, cheap print server eg. laser printer.

7. Full-generality distributed operating system.

8. Sophisticated man-machine interface.

The Programme will stimulate the UK production of hardware suitable for the 2nd Generation IPSE as described above. It is important to effect this development in a short enough timescale to prevent UK manufacturers being eclipsed by the USA and Japanese industries in this large market. Such machines will be available in 1984/5 for about £5-10K.

3.3.4 The 3rd Generation IPSE

The 3rd generation IPSE (or ISF), containing knowledge bases and 'intelligent' tools, requires significant research which must begin now if the 1989 target date for the Information System Factory is to be met.

It is envisaged that the ISF will be defined as much by market and economic realities as by any technical goals; it will (almost by definition) embody the most cost-effective ways of producing application-specific IT systems available at the time.

An Information System Factory will probably consist of six main subsystems:

1. specification and prototyping facilities

2. a Software Development Environment

3. a facility for CAD of VLSI and hardware development

4. a database or knowledge base of available software and hardware components

5. the communication systems, both local and wide area, to facilitate co-operative development

6. project management aids.

How far advanced these six subsystems are by 1989, and how closely integrated together, depends on technical advances which are hard to predict. Markets will exist for the separate components as well as the unified ISF. The following sketches are probably optimistic in their assumed rate of technical progress, but help to define the aims.

1. Specification and Prototyping Facilities

Specifications of the system under development will be held internally in a formal, machine-manipulable form (which is central to the integration of the whole ISF - since it is used by all its subsystems). There will be extensive facilities to convey these specifications to people such as system designers and the eventual users of the application system - by animating the specifications, producing small prototype systems, question-answering and so on. Completeness and consistency of the specifications will be checked automatically.

2. Software Development Environment

This will go beyond present-day environments in supporting all phases of the software lifecycle and in relating them back to the formal specifications. It will be tailored to support one of several different development methodologies - depending on the application area - and will support a defined style of project management.

3. Facility for CAD of VLSI and Hardware Development

With the emergence of special-purpose hardware architectures implemented in VLSI, the need arises for functions to migrate between software and hardware during the lifetime of an application system. So CAD of VLSI cannot be considered as a separate problem. A VLSI-implemented system must meet the same formal specifications as a software system, and pass the same tests, and vice versa. Therefore the software development and the VLSI CAD facility need to be centred around the same specification method and must communicate with one another.

4. Database of Available Components

To compete effectively in making IT application systems, it will be increasingly necessary to re-use existing software and hardware components. These components will be very diverse - a component could be a software product, an integrated circuit, a sub-routine or fragment of code, an algorithm, a man-machine interface device or one of a set of formal theories about data structures. A database of such components will hold some information common to all of them, to answer questions such as: What does it do? (i.e. its formal specification) What environment does it require? (language, storage space, power requirements, inter-connections etc) and Can it be adapted to perform a slightly different function? Some components will be general purpose and some will be application specific. Initially such information will be held in a database and searched in various ways; but in the longer term there is a need for automatic reasoning based on the data; this broadens the requirement to an intelligent knowledge based system (IKBS).

5. Communications Systems

A key feature of the ISF is the facility to allow groups of designers and programmers to work co-operatively, even when geographically distributed. This will mean a requirement for high bandwidth communication between co-operating processes, both within site, and between sites. It must be possible to implement the ISF in a distributed manner. It is expected that whilst the basic communications technologies of local and wide area networks will exist to allow this to occur, nevertheless considerable developments will be required to meet the special needs of the ISF, expecially in handling interactive high resolution colour graphics.

6. Project Management Aids

Project planning, management and control methods will be developed. When supported by a comprehensive collection of tools, these management techniques will provide both professional managers and technical staff with the ability to effectively plan and control all aspects of the software development process throughout the life cycle. These management tools must be intimately integrated into the development process to ensure that all the appropriate parameters can be realistically measured.

Thus an ISF, with all six subsystems implemented to a greater or lesser extent, will be an essential prerequisite to compete in producing medium to large scale information systems in the late nineteen eighties. It will represent a major capital investment for anyone intending to compete in the field.

The discussion so far has concentrated on the development of large, complex application systems; however, similar remarks apply equally to the small systems market. To remain competitive in producing IT products, companies will have to use advanced specification and prototyping tools, application development aids and libraries of components to produce better systems faster. So analogous small-scale Information System Factories may well dominate the small systems field, although market forces will drive them more to a low cost, high volume regime. The greater dynamism and adaptabiliy of this sector means that new approaches are always rapidly emerging and can be rapidly tested in that market. Therefore the Alvey programme will by no means ignore the small systems market; producing and supporting small scale Information System Factories will be an important activity in its own right, as well as a testbed for ideas to be used in the large-scale systems market.

3.3.5 Concluding Remarks on Integration

Thus the strategy for producing the three generations of IPSE requires a controlled set of concurrent and overlapping research and development activities. It is important that the 1st and 2nd generation IPSEs are produced, not just the 3rd generation ISF, because major gains are expected in software productivity and quality from their UK installation and exploitation as well as export sales.

3.4 Innovation

The Director (SE) will initiate a programme of research to ensure that the scenario in section 2 (The Changing Nature of Software Development) is realised in the UK. This will require a balanced programme of directed contract work and responsive funding. The universities will play a significant part in this work and the SE programme expects to work closely with other funding bodies and initiatives. The Director (SE) will sometimes let competition develop between research teams as well as organising collaborative projects, some between companies and some including universities as well.

The SE research programme will overlap significantly with other areas (this is a good thing) and the Alvey Directorate will ensure intra programme coordination.

The three key points to be made about innovation are that

i whilst the general directions in which innovation is needed are known it would be premature now to try to pick winners and ignore rival approaches;

ii research projects are often on too small a scale to provide an adequate testing ground for a new technique;

iii the scale of UK research must be increased to compete with our international rivals.

Thus the programme will back a number of promising approaches to (for example) specification, and test them out on life-size projects rather than attempt to evaluate them in terms of their apparent success in small-scale use. This approach not only offers a better chance of selecting useful techniques, it also starts to bridge the 'development gap' by bringing research results out into a development environment.

The current list of research priorities includes:

i. Software Development Methods

- Formal Specification
- Verification and Validation
- Reusable Components
- Metrics
- Quality Assurance and Certification

ii. Project Management

- planning and estimating
- progress and productivity measurement
- budgeting
- standards control

iii. IPSE

- items already indicated above are relevant
- evaluation experiments to test changes in productivity and quality due to use of IPSE in the industrial context
- MMI, VLSI/CAD etc from other Alvey areas but relating to IPSE construction

The list of research priorities will be regularly reviewed and, if necessary, modified. In addition to the above list which sketches out some of the work required to achieve the programme's major goals and objectives it will also fund a small amount of longer term and/or more fundamental research to maintain a balance between targetted development and pure research. The theoretical underpinnings of software engineering are considered to be of vital importance - a thorough 'understanding' must precede the expensive construction of the sophisticated ISF-type environments.

3.5 National Quality Certification Centre

The primary medium term payback activity is seen as the creation of a National Quality Certification Centre (NQCC) for software products and components. The NQCC must build up an international reputation. This will involve the adoption of state of the art techniques on a continuous basis. The commercial benefit of NQCC approved software products in an international market is potentially extremely valuable. As the mass market for software products develops consumers will buy NQCC approved products rather than unapproved products. The rapid establishment of such a national capability could give the UK a significant commercial advantage.

The concepts behind the NQCC are currently in their infancy with only communications protocols and programming language compilers being 'certified'. The NAG library quality control reputation shows the potential benefit of extending this concept.

The NQCC should provide a realistic focus for much speculative research and development work.

The NQCC cannot spring into existence overnight. It is envisaged that early in the programme one or more R & D centres will be established to develop quality assurance and certification techniques. At least one centre's medium term aim will be to transform itself into the Alvey Quality Certification Centre. If the AQCC can establish a national reputation then the move to genuine 'national institution' status, possibly as an independent, revenue earning body, should rapidly follow.

3.6 Software Production Centre

The SE programme will establish a Software Production Centre. This will not be a research project but a working factory funded to exploit and incorporate the latest technology. The facilities of the centre will be made available to software producers for genuine production work.

This will enable large organisations to try out 'real' new techniques before making the necessary in-house investment. It will also enable small companies to experience the benefits of new technology which they could otherwise never afford.

The SPC will be aimed at producing software which will pass the tests laid down by the National Quality Certification Centre.

Technically, the SPC is to be a multi-lingual, database foundation, integrated project support environment. It will act as the focus for the practical embodiment of much research and development work.

It will support not only the development of new software but the maintenance and evolutionary development of existing products. To this end, it will be 'multi-lingual', ie it will be capable of developing systems in, say, Cobol, Fortran and Coral as well as eventually, say, Ada and Prolog. It will also be 'multi-lingual' in the sense that any one of its software products can be constructed from components coded in several different languages. Such a requirement will stimulate the development of re-usable software components and maximise the return on investment in existing software.

The technology contained within the Software Production Centre could be exported into the sites of the participating organisations by:

a. replication of hardware and software components on the site

b. network access from the site to the Centre

c. a combination of a and b.

The running of both the NQCC and the SPC will be contracted out to industry.

3.7 Development Programme

The NQCC and Software Production Centre require a research and development programme to feed them with new technology. The Director (SE) will initiate a medium term R & D programme to ensure that the goals given in section 1 are realised throughout the UK. This will require a balanced programme of directed contract work and responsive funding.

3.8 Software Components Brokerage

The Director (SE) will examine the desirability and feasibility of a centre for software components and products. It will operate by holding specifications, code etc in a database accessible only via the Alvey network. Dissemination of components will be only by FTP (file transfer). Participants will lodge their components and products in the database with distribution at a charge. This scheme should encourage collaboration, technology transfer and the creation of reusable software components and products, the idea being that it will be quicker and cheaper to get a subroutine from the brokerage than to reinvent it.

Two types of software products are envisaged as being handled by the Brokerage:

a. Packages for sale to the public. These should ideally have been approved by the NQCC.

b. Reusable software components. These too should ideally have NQCC approval but will not be on sale generally. They will be available to the 'trade', ie to those software developers who will attain increased productivity by using existing components rather than developing their own and who will contribute components of their own manufacture. The Software Production Centre should be a major source of, and customer for, these components.

With seed money from the Alvey programme to assist its launch this should become a commercially viable operation.

3.9 Product Stimulus

Industry must produce products. The sales of such products are one important evaluation criterion for the Alvey Programme, other government initiatives and the health of the industry. However, short term sales figures will not be the dominant factor for Alvey Programme assessment.

The programme will work collaboratively with other industry and Government initiatives, such as the Software Products Scheme, to ensure a smooth transition from Alvey-supported research into more market orientated activities. This will help to avoid the creation of an Alvey development gap. Conversely, the programme will welcome input from such initiatives which perceive market pressures having implications for the programme's strategy and priorities.

4. PLAN

For Immediate Action

The following programme activities will be initiated within one year of inauguration of the programme.

1. For the short term, UNIX will be used as a basis (but not a standard) for the 1st Generation Integrated Project Support Environment (IPSE).

2. The Programme will encourage the development of compatible sets of today's technology tools for UNIX, together with associated training and operational support material. Ideally the tools will be formally specified, efficiently implemented on UNIX and capable of being efficiently reimplemented on other hosts. The feasibility of producing standards for software tool interfaces will be investigated.

3. The Programme will fund installations of the 1st Generation IPSE with tools developed under (2) -

   i in selected organisations, on condition that those organisations (a) invest in training and use of the installation and (b) provide data for the measurement activities (7).

   ii in 'Centres', to be available for experimental evaluation and for genuine production work; installations in 'centres' will support access from remote sites across the network.

4. Via the Alvey Network establish procedures and systems for dissemination of information and reporting appropriate to the Software Engineering programme and the 1st Generation IPSE base.

5. Encourage (and fund, where appropriate) other (ie non-UNIX) implementations of tools and additional educational material, to the same specifications as developed in (2).

6. Establish methods of co-operation and information interchange with Ada APSE development activities.

7. Establish a programme of investigation of measurement methods for quality and productivity.

   i Commission analyses of industrial practice, with measurements of the development of real systems.

   ii Commission studies of the measurement techniques that could be applied and how they would be implemented.

Less Immediate Payback but still Short Term Initiation

8. Plan and initiate a comprehensive programme of R & D for the development of the 2nd Generation IPSE. The aim will be to take today's research results and apply them in practice. Early components of this activity are the following:

   i Identify candidate techniques in the areas of

   - specification
   - transformation methods
   - verification methods
   - validation methods
   - design methodologies
   - system construction tools
   - testing methods,
   - metrics and quality assurance

   ii Commission work applying the candidate techniques to life-size projects, in competition with rival techniques.

   iii Compare the results of the competitive analyses from the viewpoints of

   - feasibility
   - scope
   - demonstrated improvements in quality
   - cost of application
   - etc .....

   iv In co-ordination with the above, analyse information requirements through the development life-cycle so as to lead to

   - integrated development methodologies
   - requirements specification of 2nd Generation IPSE database facilities.

   v Commission relevant human factors research in association with the Alvey MMI programme.

   vi Develop hardware and software for the 2G IPSE.

9. Support relevant research activities not immediately linked to development or production goals where additional funding is necessary. Areas include:

   - logic programming and logic databases

   - programming language theory

   - nature and specification of interfaces between system components in multi-lingual or mixed hardware/software systems

   - techniques for exploiting concurrency.

Initiate Medium Term Payback Activities

10. Set up and operate three major 'centres of activity':

- National Quality Certification Centre
- Software Production Centre
- Software Components Brokerage.

5. PROGRAMME MANAGEMENT AND INFRASTRUCTURE

5.1 Introduction

Consultations have established that there is widespread agreement throughout the academic and industrial communities on the technical issues of

a. the nature of the software crisis
b. the likely technological progress (excluding any revolutionary breakthrough)
c. the research and development needed to achieve this progress.

This section concentrates on the management and organisational aspects of what the Alvey Programme must do in practice to achieve substantial progress in

a. increasing the quality of software products
b. reducing production and maintenance costs
c. remaining competitive in an international market.

5.2 SE Programme Management

The Alvey Director has appointed a full time Director (Software Engineering), Mr David Talbot, to run the Software Engineering programme.

The Management of the Software Engineering Programme will follow the main recommendations of the Alvey Report. It will

(1) be a directed programme. The Director Software Engineering will be responsible and accountable for the decisions made, within the structure of his responsibility to the Director of the Alvey Programme and the Steering Committee

(2) be a few in number. It will be staffed by a small number of skilled professionals with strengths and capabilities suited to the pursuit of the three main objectives of the Work Programme - exploitation, integration and innovation. These will be drawn from Industry, academe, SERC and MoD.

(3) as a result, aim to work through others. Specifically it will use, as far as possible, existing Government advisory mechanisms to assist in the shaping of the overall strategy; the achievement of better levels of coordination and the promotion of support for the strategy. This will be achieved through the use of the EARB Computing & Communications Committee as the main software engineering strategy advisory group. Furthermore the Director will seek to establish specialist panels to assist in the development of programmes for the specialist topic areas. These panels could also form the core of Alvey Software Engineering "special interest clubs".

(4) normally let contracts on a collaborative partnership basis with one contractor responsible for the appointment of a coordinating manager for the project, responsible for its administration and execution. Guidance Notes for potential participants are in the course of preparation.

(5) sponsor collaboration. It will, on suitable occasions, be regarded as a part of the management process to suggest partners for projects. Final decisions on this will at all times, however, rest with the proposers themselves.

(6) recognise the importance of smaller companies in the software industry. In particular it will seek to evaluate the best way to help them and will take suitable action if smaller organisations' needs differ from their larger counterparts.

Overall the management aim will be to seek strong general support for the strategy, adherence to it as a key criteria for the selection of projects and the establishment of a sense of continuity within the programme.

5.3 Co-ordination

Co-ordination and direction of the SE programme are vital activities. Given the wide spread of other bodies active in this area 'external' co-ordination is also very important. At present, there is a lack of co-ordination between various funding bodies.

Co-ordination can only be achieved by technically competent staff travelling around talking to people 'at the coal face'. This activity will be backed up by the creation of Special Interest Clubs (a la BCS), workshops, conferences, mailshots and electronic bulletin boards, using the programme's communications network.

5.4 Education and Training

There is widespread agreement that education and training are vital activities. The Director (SE) will formulate an education and training requirement based on his continuing co-ordination experience. He will input this requirement to the appropriate bodies to ensure the programme's needs are met.

The likely widespread increase in the use of mathematically based techniques, such as formal specification and quality certification, will only be profitably exploited by UK industry if staff (young and old) have been trained in the basic principles of formal methods. The programme will therefore encourage the increase of education and training in this area.

5.5 Recruitment & Staffing

The SE programme will be limited by the shortage of skilled manpower. The Director (SE) will take steps, in addition to education and training, to ensure that the UK retains its good people and actively entices people who have already gone down the brain drain back to the UK. Rapid, flexible decision making and project funding are likely to be key factors in preventing disillusionment fuelling the brain drain.

5.6 Creating a Community

The Director (SE) is responsible for breaking down the barriers between academia and industry to create a genuine community which will work co-operatively. Breaking down the barriers will involve:

a. forming person-to-person links

b. providing good communication channels (workshops, network mail, newsletters, etc)

c. funding collaborative projects which deliberately cause new links to be forged.

Such activities as the 'dissemination of existing knowledge' education programme, the first generation PSE and tool kit should also help to create a community.

5.7 Planning and Evaluation

5.7.1 Planning

Planning will be done on a continuous basis, both from the short term and long term viewpoints.

5.7.2 Continuity and Consistency

The policies and plans will be consistently implemented so that good personal relationships and understanding can be built up between the Directorate and its participating clients. Stop-go, chop and change attitudes will be counterproductive.

5.7.3 Evaluation

A serious, continuous attempt will be made to evaluate the impact and benefit of the SE programme. This will not be easy but is vital.

5.8 Infrastructure

The whole Information Technology industry, not just Software Engineering, is a major user and developer of software, and so the degree of ease with which software can be developed and shared affects the productivity of many people.

5.8.1 Academic Software Engineering Research Infrastructure

Currently the academic software technology base is very non-uniform in that the knowledge, experience, tools, techniques and equipment vary considerably between individual projects. SERC's Common Base Policy (see Appendix 2), which applies to all SERC funded scientific and engineering subjects not just information engineering, aims to raise the overall technology level and make it uniformly higher by bringing together good tools, packages and techniques into a uniform framework.

It is proposed that SERC's existing Software Technology Initiative Infrastructure, which contains the SERC Common Base Policy, be adopted for the academic research component of the Alvey SE Programme thereby building up the 1st generation IPSE. (Details are given in Appendix 2.)

5.8.2 Industrial Infrastructure

The SERC Common Base Policy has 3 components - software, hardware and networking. Is it feasible or sensible to propose an industrial equivalent?

Common Software Base:

Industry uses a wide variety of languages and operating systems so no natural commonality exists. All that can be done in the short term is to encourage the use of UNIX amongst those organisations who wish to benefit from the 1st generation IPSE activities, the SERC Common Base Policy and its associated academic research.

Further consultation will take place on the industrial software infrastructure, especially with a view to the construction of the 2nd generation IPSE.

Common Hardware Base:

The SERC has chosen the ICL PERQ powerful single user computer system as its first hardware base. This kind of equipment is seen as the way forward for IPSEs. However at this stage in the Programme it is not possible to define a short term hardware base for the industrial component of the Programme. Many of industry's existing computers are capable of running UNIX. In the longer term new hardware will be developed for the 2nd generation IPSE so it is too soon to make recommendations.

Common Network Base:

This is a vital development to link people and machines into a working community and is treated separately as part of the overall Alvey programme.

5.9 Relationship with ESPRIT

The main Alvey Report identified that the UK could not regard ESPRIT as a substitute for a strong UK programme; rather a UK programme would help the UK participate more effectively in the European one. The Alvey software engineering programme fully supports this view. It will seek at all times to identify how best to establish good levels of synergy between the two programmes and in particular the Alvey programme will aim to establish criteria that will allow it to determine where it will need to run totally parallel activities; where it would suggest an Alvey led programme; where it would be content to build upon ESPRIT and where it would be prepared to have ESPRIT handle a complete area of activity. In establishing these criteria the Alvey Directorate would seek wide consultation within the community. It would anticipate, however, that the following factors would play a significant part in the decision making process

- Criticality of topic area. Can the UK leave it for others to do? Does the UK have an important dependancy?

- Timeleness of proposed programmes

- Transferability of results

- Need for more than one programme in the area considered

- Weight of expertise in topic area

Areas which at this stage suggest that close cooperation could be mutually beneficial would be standards, distributed UNIX, metrics, specification and verification methods.

5.10 Concluding Remarks

The UK currently has an established and vigorous software sector in the UK with a mushrooming growth in the number of companies offering software products and services, significant overseas revenues and an academic community of world standing. In spite of this the UK continues to have an adverse balance of trade and its ability to maintain or, indeed, strengthen its position in this lucrative but intensely competitive market area will come under increasing pressure, not least as a result of the major programmes currently underway in its competitor countries.

An ability, therefore, to establish a capability of producing cost effective, high quality software is an essential pre-requisite to the continuing health of the IT industry and, indeed, to the rest of UK industry which is already dependent on the provision of world class computer systems if it, in turn, is to maintain and develop its competitiveness. Consultation in all sectors of the community has identified that

- the increasing dependence of business on economic, reliable and soundly designed computer systems

- the growing complexity of such systems

- the general failure of software producers to meet users' needs in terms of timeliness, cost and quality

- the specific programmes already in hand in America, Japan and France

all point to a critical need to discard the present ad-hoc 'craft' practices of software production and to ensure that UK managment turns increasingly to capital intensive methods of efficient software production and an engineered approach to reliability and quality. It has been with this goal in mind that the programme proposed in this document has been devised.

In setting out to meet the above it is accepted that many important areas of interest have been omitted, eg database development, except for those aspects needed to establish distributed IPSEs, and the production of specific applications such as CIM. In doing this the Alvey programme has taken the view that it has finite resources and that these need to be focused in the sharpest manner on the area agreed as of fundamental underlying importance. This being said, the Alvey Directorate will nevertheless aim to keep the areas of omission under regular review with a view to their possible future inclusion in the programme.

6. ACKNOWLEDGEMENT

The Alvey Directorate gratefully acknowledges the widespread cooperation and assistance of the many people from industry, academia and government who have helped formulate the Software Engineering strategy.

Particular thanks are due to PH, JT, BW, VS, LV and JG who made direct contributions to the document.

PART II APPENDICES

1. SERC STI INFRASTRUCTURE

2. SERC COMMON BASE POLICY

APPENDIX 1

SERC SOFTWARE TECHNOLOGY INITIATIVE

INFRASTRUCTURE POLICY

1. SUMMARY

The STI infrastructure plan consists of equiping a typical STI group with a Multi-User Mini (MUM) acting as support to a set of Single User Systems (SUS). The SUSs used will be in line with the Council's Common Base Policy (CBP). Both MUM and SUSs will run the CBP common software base and be linked together by CBP network equipment and protocols.

2. INTRODUCTION

The Software Technology Initiative is establishing an infrastructure, containing the SERC's Common Base Policy (CBP), which will form the framework within which to research and develop the first and second generation IPSEs (integrated project support environments) as outlined in the STI's research policy and the Alvey Report.

In outline the SE Infrastructure envisaged will consist of a powerful, multi user mini (MUM) Unix system linked by high speed local area network to a set of high performance Single User UNIX Systems (SUS) running the Common Software Base and Communications protocols.

In more detail the SE Infrastructure envisaged will comprise:

a.	SE Research Tools (examples)	ML, HOPE, LCF.	
		Ada, Modula-2.	
		Affirm, Stanford Pascal Verifier.	
		Boyer-Moore, Iota.	
		LISP	(IKBS recommended version)
		Prolog	(IKBS recommended version)
		Pop-2	(IKBS recommended version)
b.	Languages	Pascal	(ISO Standard)
		Fortran 77	(ANSI Standard)
c.	Graphics	GKS	(BSI and draft ISO Standard)
d.	Operating System	UNIX	(32 bit, virtual memory)
e.	Computers	32 bit SUS	(current CBP m/c is PERQ)
		32 bit MUM	
f.	Servers	(Not yet available)	
g.	Local Area Network	Cambridge Ring	(UK CR82 standard)
		Ethernet	(when approved by CBP)

h. Wide Area Network X25 (SERCnet & PSS)

i. Protocols JNT Coloured Books (UK academic standards)

The following gives a more detailed exposition of the technical components and philosophy of the policy.

3. SE RESEARCH TOOLS

The following list contains those tools which are known to be of interest to several groups of researchers and therefore would benefit from wider availability via the SE Infrastructure. The list should be regarded as 'provisional'.

3.1. Functional Languages

ML, HOPE, LISP.

An STI workshop at RAL led to the expression of considerable interest by the community in having LISP, ML and HOPE running identical versions on the PERQ, VAX and STI MUM.

ML is associated with LCF (see below).

3.2. Logic Languages

PROLOG

There was some interest expressed at the RAL workshop in PROLOG. The STI programme should be able to get PROLOG via the IKBS programme.

3.3. Procedural Languages

ADA, MODULA-2

Ada is available on VAX/UNIX from the STI funded work at York. The same compiler is being moved to the PERQ.

The Cambridge VAX/UNIX Modula-2 compiler is being moved to the PERQ.

3.4. Verifiers, Theorem Provers

LCF, AFFIRM, SPV, B-M, IOTA

The UK does not have all of these systems. Only LCF and Boyer-Moore are available, currently on the Edinburgh DEC-10 which will be shutdown in October 1984.

There is considerable support for having LCF available on the VAX, the MUM and PERQ.

The Stanford Pascal Verifier is being mounted on the PERQ by Strathclyde via an STI EMR.

The STI Coordinator and Prof Jones, Manchester are working at obtaining AFFIRM and IOTA.

4. LANGUAGES

- see CBP Appendix.

5. GRAPHICS

- see CBP Appendix

6. OPERATING SYSTEM

- see CBP Appendix

7. COMPUTERS.

7.1. Single User System

The recommended CBP PERQ configuration is

PERQ:
1 Mbyte main memory
16K writeable control store
24 Mbyte disk
tablet
puck (3 button 'mouse')
LAN interface
X25 front end (one per installation)
1 Mbyte floppy
100 Pixel per inch A4 display

For further details see CBP Appendix

7.2 Multi User Mini

The IPSE requires not only high performance SUS for highly interactive tools but also major computational power (verifiers), storage (database, backup) and peripherals (printer, communications, tape deck, archiving). It is proposed that each research department engaged in a significant amount of SERC SE research have a MUM to complement the SUSs. The MUM must run the same Common Software Base as the SUS. The MUM is a necessary infrastructure component because

a. STI investigators need to run imported software

b. a MUM allows more users to be given access to simple services which do not require a PERQ.

c. PERQs need a host machine to provide an interim 'server' capability for archiving, printing etc

d. many STI investigators now require computationally intensive tools such as theorem provers especially the VAX/UNIX implementation of LCF from Cambridge

e. it is important, as far as is possible, to provide technical compatibility with the IKBS community as productive cross fertilisation is likely

f. an infrastructure machine could provide a pump priming facility to enable work to be done prior to and in preparation for an SERC grant.

The envisaged SE Infrastructure MUM configuration is

MUM: Powerful, multi-user minicomputer eg GEC series 63, VAX
4-8 Mbytes main memory
500-1000 Mbytes disk storage
9 track 1600 bpi tape deck
Line printer, upper and lower case
Documentation quality printer (till Laser Printer LAN server)
Local area network connection
Wide Area network connection (till LAN server)
5 'conventional' VDUs (+5 PERQs)

7.3 Linked SUS-MUM IPSE Distributed Computing System

The MUM/UNIX system is envisaged as being accessible to 5 users via conventional terminals and to 5 users via high performance SUS PERQ/UNIX systems, via a high speed local area network. Peripherals, including a laser printer and wide area communications, will be shared by all (SUS + MUM) systems through LAN servers eventually and via the MUM in the short term.

Such a distributed computing system will consist of 6 UNIX systems connected via file transfer and remote login facilities based on the JNT coloured book protocols. It is envisaged that a more unified, logically integrated (remember the IPSE target), physically distributed UNIX software environment will come into operation in the mid 80s as a result of non UK work, the SERC's DCS Programme and general UK progress towards the IPSE.

8. SERVERS

- see CBP Appendix.

9. LOCAL AREA NETWORK

- see CBP Appendix.

The STI Panel would like to see the CBP evolve to include an Ethernet type of LAN

10. WIDE AREA NETWORK

- see CBP Appendix.

11. PROTOCOL STRATEGY

- see CBP Appendix

12. PORTABILITY

- see CBP Appendix.

13. GENERAL POINTS

- see CBP Appendix.

14. REFERENCES

- see CBP Appendix.

APPENDIX 2

SERC COMMON BASE POLICY

1. INTRODUCTION

1.1 Overview of Distributed Interactive Computing

The appearance in the market place of cheap high powered single user computer systems with good interactive capabilities via high precision displays, linked together by high speed local area networks, heralds a completely new way for most SERC Investigators to achieve the major part of their computing requirements.

Within the next few years, many such systems will be available from different manufacturers. Consequently there is a likelihood of many different systems being purchased in the SERC environment leading to a great deal of duplication of basic software development.

SERC sees a need for a coordinated development plan to ensure that the UK makes the best use of its finances and of its limited manpower. The SERC has therefore decided on a strategy of creating a common hardware and software base for software development which will encompass all scientific subject areas. Briefly the common software base will be Pascal and Fortran running under the Unix operating system implemented on the common hardware base of PERQ single user computers linked locally by Cambridge Rings and nationally by the X25 wide area network systems (SERCnet and PSS).

SERC Subject Committees will participate in the implementation of this policy by enabling central purchasing of PERQs for grant holders to be done via Central Computing Committee and by ensuring that investigators use the PERQ in all appropriate circumstances as well as encouraging them to follow the common base software development policy. The Common Base Policy is not the same as standardisation, however, and it will evolve as the state of the art improves.

1.2 Common Base Policy

The whole academic community, not just Computer Science, is a major user and developer of software and so the degree of ease with which software can be developed affects the scientific productivity of many researchers.

The SERC has approved a plan to increase the productivity of scientific research requiring computing by:

(1) facilitating scientific cooperation by:

- (a) person to person links
- (b) computer to computer links
- (c) common software and hardware base policy.

(2) Set in motion a coherent plan to exploit software tool production by making such tools/techniques widely known and available in forms which can be readily used by the whole user community.

Currently the academic software technology base is very non-uniform in that the knowledge, experience, tools, techniques and equipment vary considerably between projects. The motivation to create a common Hardware and Software Base is to bring together all of the best existing tools, packages and techniques into a uniform framework so that the 'whole' is more effective than the 'sum of diverse parts'. This will be achieved via EMR contracts to move existing software into the common base, specific purchases, the direct results of SERC research projects using the common base equipment and the 'snowball' effort that will be generated as a natural consequence of providing a state of the art hardware base. A good example of the common base 'snowball' effect is the widespread use of the Unix operating system which has enabled a large number of software tools to be made available throughout the UK academic community.

The Common Base Policy briefly is:

(a) common software base,

(b) common hardware base,

(c) common communications.

The SERC wish the common software base to be the Unix operating system and the common hardware base to be the PERQ. The PERQs should be networked together via Cambridge Rings, SERCnet and PSS to allow widespread cooperation between users and developers. This combination of software and hardware is widely accepted as being the best combination for developing software in the coming years. A common base does not imply rigid standardisation however.

Computer technology develops at a rapid pace and it is expected that the next few years will see the cost of single user systems decline and their quality and capability increase. Therefore today's PERQ is seen as only the first machine forming the common hardware base. The common base will develop over the coming years.

1.3 Outline

In outline the Common Base Policy comprises

a. Pascal (ISO Standard)

b. Fortran 77 (Ansi Standard)

c. GKS (BSI and draft ISO Standard)

d. UNIX (32 bit virtual memory - de facto standard)

e. PERQ (High performance single user system)

f. Cambridge Ring (Local Area Network)

g. X25 (Wide Area Network)

The following gives a more detailed exposition of the technical components and philosophy of the policy.

2. LANGUAGES

2.1 Pascal and FORTRAN 77

Pascal and Fortran 77 have been chosen as they are the two most popular scientific languages. They possess the properties of portability and official standard definitions. There is a large amount of software already written in them which allows people to make use of existing investment.

There will be considerable SERC support for Fortran 77 and Pascal. This will take the form of software tools and techniques developed by the Software Technology Initiative and the activities of the SERC Computing Service team. Thus the CBP will act as a focus for many different activities.

The technical definition of Pascal is given in (ref 1).

The technical definition of Fortran 77 is given in (ref 2).

2.2 Other languages

Other languages will be available with the set of software tools in the CBP. For instance the Unix 'C' language is already available and Ada is under development by York. LISP and Prolog are being implemented.

These other languages will not receive the same degree of support and tool development as Pascal and Fortran. They are not 'blessed'. This situation must be reviewed regularly. Specific minority groups eg Ada community will receive minority support through individual committees eg STI.

Evolution of status from 'other' to 'blessed' is possible.

2.3. Mixed Language Working

It is a requirement of the CBP that 'blessed' languages should be inter-workable at the procedure call level ie a Pascal program can call a Fortran subroutine which can call a Pascal procedure etc. This is a vital capability to ensure maximum use of standard components. It is ridiculous to have to, say, reimplement a Fortran graphics package in Pascal because Pascal cannot call Fortran.

Interworking has implications for compiler construction and operating system development. It has its limitations and difficulties, eg the difficulties in enforcing type checking across procedure interfaces, but its benefits outweigh its drawbacks. (Reference Tony William's paper).

3. GRAPHICS

In line with the policy of supporting international standards and portability aids the CBP has 'blessed' GKS 7.2 as its basic graphics package. GKS will be available on all SERC machines, not just PERQ, to help the transfer of graphics software and, via metafile standards, pictures themselves.

There will have to be a significant amount of software mounted on top of GKS to give the scientist the graphics facilities he requires. Much of this graphics library porting work will be led by RAL Graphics Section.

The technical definition of GKS is given in (ref 3).

The technical definition of metafile standards is under development by ISO.

4. OPERATING SYSTEM

4.1 CBP UNIX

Unix is already a de facto standard in many academic institutions in both USA and UK. It has enabled a great deal of software to be shared amongst research groups and has built up a large quantity of widely applicable software.

Unix is being used increasingly by industry again both in the USA and UK. The CBP philosophy is based on the following properties of Unix.

a. It is popular ie a de facto standard.

b. It is implemented on a wide variety of makes and sizes of computer (IBM 370 - M 68000).

c. It is manufacturer independent.

d. It is cheap ($150 per PERQ).

e. It has a large body of user level software.

f. It is used by both industry and academia.

For the scientific community Unix is likely to become the standard small machine operating system because 'small' machines seem to get bigger every day!

The CBP Unix has the following properties

a. It is full '32 bit'.

   Arithmetic is 32 bits by default to overcome the annoying limitations of microprocessors. 8,16,32,64 bit quantities are available.

b. It is virtual memory.

   Full 32 bit addressed linear address space (via paging) removes the size restriction which is often so frustrating.

c. CBP Unix is System III.

The technical specification of Unix is given in (ref 4).

4.2 UNIX Evolution

There are several versions of UNIX either in existence or soon to be announced. These include Berkeley 4.1 and 4.2, Bell version 7, System III and System V.

The CBP philosophy is to run the same, stable version of UNIX on all the different types of hardware supported by the CBP ie only one version of UNIX will be supported by SERC.

There must be a balance between the benefits of new developments and the benefits of stability and standardisation. Thus moving to a new version of UNIX will be a major evolutionary step for the CBP, especially if and when more than one CPU type is involved.

5. SINGLE USER SYSTEM

The PERQ was and is the first machine which satisfies the requirement for a high performance single user system (see Appendix A). Other machines are likely to follow (some are already here). The expected proliferation of machines will tend to fragment the software development activities because some things will always be machine specific. The Council therefore wishes to balance the benefits of standardisation (which acts against change) with the need to give state of the art facilities to scientists (which requires change). The future CBP is therefore expected to include more than just today's PERQ but such changes must be taken infrequently and given very careful consideration beforehand.

It should be borne in mind that the criteria for choosing a single user system must be that it runs the common software base rather than has some new hardware feature. The investment in software is already so large that computers must be purchased which run the Council's software rather than the Council's money be wasted on reimplementing existing software on some new hardware. Manufacturers will have to understand the changing balance of power between them and their customers. The manufacturer independence of Unix is a key factor in this equation.

The recommended CBP PERQ configuration is:

PERQ:
- 1 Mbyte main memory
- 16K writeable control store
- 24 Mbyte disk
- tablet
- puck (3 button 'mouse')
- LAN interface
- X25 front end (one per installation for connection with WAN)
- 1 Mbyte floppy disk
- 100 pixels per inch A4 display

For advice on peripherals such as printers suitable for use with PERQ contact CBP User Support at RAL.

The technical specification of the PERQ is given in (ref 5, ref 6).

6. LOCAL AREA NETWORK

6.1 Cambridge Ring

The CBP requires a fast local area network to link its machines together. The Cambridge Ring has been chosen because it is

a. a UK draft BSI standard (CR82, ref 7).

b. DCS Programme's common equipment

c. has protocols already implemented for Unix which are a de facto UK academic standard.

d. much greater installed base in the UK than 10 MHz Ethernet as UK universities through their own efforts, together with DCS and JNT, have installed more than 20 Rings already.

e. it is an easily purchased and maintained commodity from a variety of UK suppliers.

The technical specification of the Cambridge Ring is given in (ref 7).

6.2 LAN Evolution

The Cambridge Ring is not the only LAN currently available, but has been chosen as the CBP LAN for the above reasons.

There are several different types of Ethernet and Token Ring LANs available or soon to be announced. The IEEE 802 standard initiative is having a beneficial influence but has yet to be adopted as an ISO standard.

The CBP will therefore stay with the Cambridge Ring and its associated CR82 protocols until the world wide LAN developments have stabilised sufficiently to enable an evolutionary step to be made.

6.3 Campus X25 Switches

Where a campus has installed an X25 system to act as a LAN then the SUS can access this via the hardware and software given under section 8, ie X25 campus LANs are 'blessed' by the CBP.

7. SERVERS

The long term objective of the CBP is to exploit the advantages of distributed computing and LANs which can be realised as Servers. The following Server requirements can be identified as desirable but not yet deliverable as 'service' equipment. There is an urgent need to develop such servers into commercial products.

7.1 Publication Quality Printing

There is a requirement for sophisticated, high quality (at least 300 pixels per inch) text and graphics printing capability to complement the Single User System's display.

Examples are hardcopy of scientific papers (camera ready including diagrams), graphical software tool output, 'mathematical' text (proofs) and so on.

It is envisaged that this need will be met by small, relatively cheap (£10K) laser printers, one per department, configured as a LAN server. Until this technology is readily available (1984?) such items as Diablo daisy wheel printers and Versatek graphics devices are suggested (Contact RAL CBP team for advice).

7.2 LAN/X25 Gateway

It is seen that an LAN to X25 (SERCNet and PSS) gateway will be the most cost effective way of connecting a number of machines to the WAN. No products are currently available.

7.3 LAN/LAN Gateway

Multiple campus LANs, whether the same type or not, are likely to arise with the consequential need to connect one LAN to another.

No products are currently available.

7.4 File Server

Single user systems cannot hold all of the data to which a single user requires access. Nor can a SUS handle file backup and archiving requirements.

In the short term the CBP recommends that SUS are not used stand-alone but are connected to multi-user machines with suitable peripherals to allow file access and archiving.

The more desirable solution is to have file and/or archive servers. No products are currently available.

8. WIDE AREA NETWORK

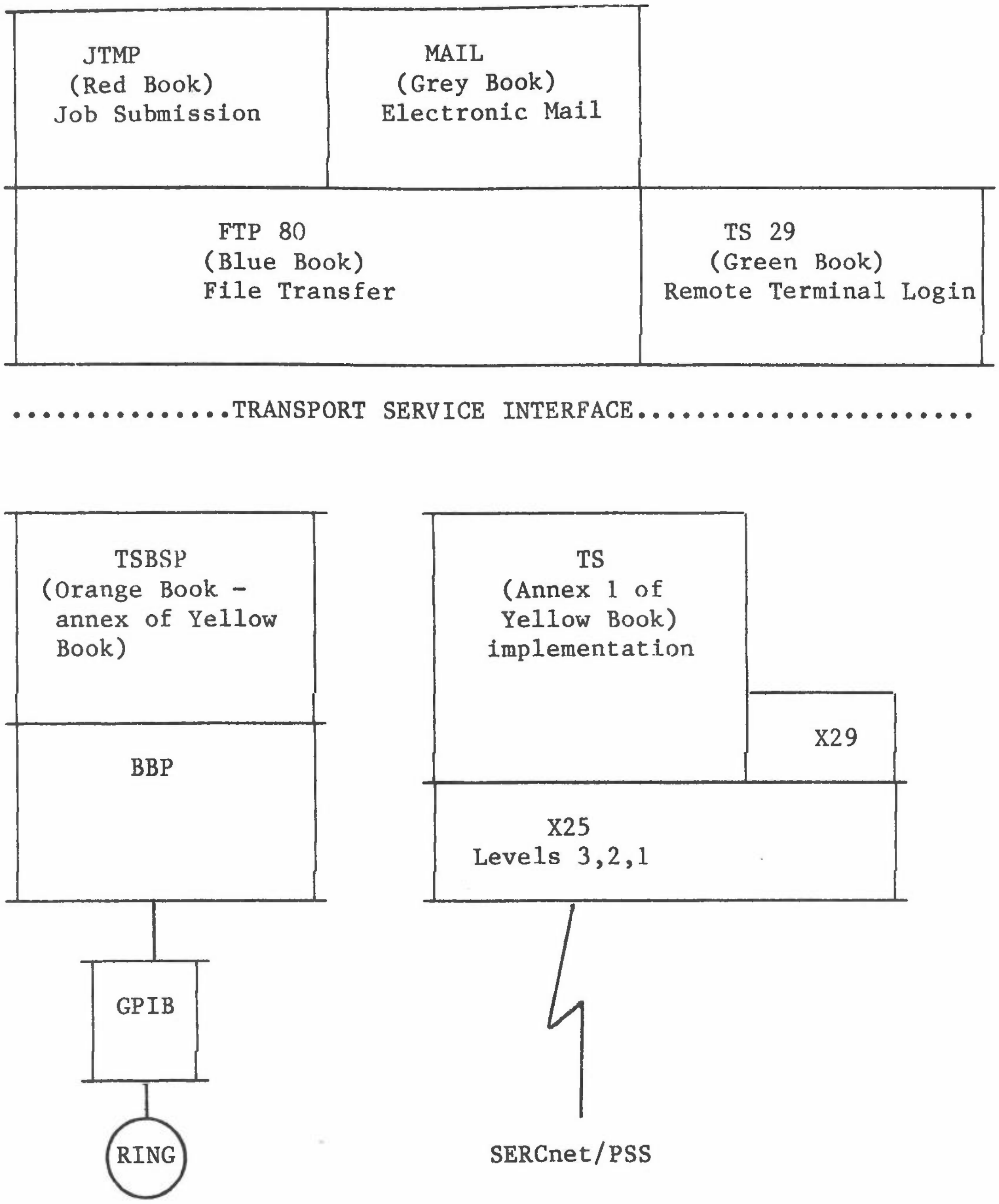
The CBP requires a national wide area network to link both people and machines. The network must be compatible with JNT/NMC policy. The current CBP uses SERCnet and PSS which are technically compatible X25 networks linked by a gateway.

The CBP also requires access to Europe (including Scandinavia) and the USA. Such links are not all easily available.

The PERQ-X25 connection, in the short term, will be via the York LSI-11 transport service front end originally designed for the PDP-11. Studies are in hand for 'in-board' solutions.

The technical specification of SERCnet X25 network is given in (ref 8).

9. PROTOCOL STRATEGY



9.1 CBP Protocols

The protocol strategy is based on the de facto UK academic standards approved by the SERC/CB JNT in their 'coloured books'. The adoption of the Wide Area Network protocols of transport service and above for the local area network use gives a useful unification of LAN/WAN facilities. The average user sees only one and the same mechanism to move files, mail etc between machines independent of distance (ie local or wide area net). The adoption of transport service also gives a degree of hardware independence for local area networks.

The use of wide area protocols for local area networks is 'conservative' in that it does not allow various advantages of LANs to be exploited eg speed, reliability. More LAN specific (light weight) protocols could be employed for high speed intermachine interaction (eg remote process execution). Such protocols should only be 'blessed' if they attain a measure of widespread acceptability. Specific research projects are likely to require lightweight protocols. They should not be discouraged in appropriate circumstances.

Transport Service around the Ring is implemented by TSBSP (Transport Service Byte Stream Protocol) running above BBP (Basic Block Protocol). These are the de facto UK academic Ring standard protocols based on Cambridge University's work.

Currently the JNT is having the Mace box built by Orbis which will be a high speed intelligent interface having TSBSP and BBP in it so providing a DMA transport service to its host.

The protocols specifications are given in (refs 9-15).

9.2 Conferencing, Bulletins

Electronic Mail as implemented over the Grey Book is an extremely useful facility. However, experimental work at various sites in the world has shown the potential advantages of more sophisticated facilities above simple mail. Such facilities include message based conferences and public electronic bulletin boards.

No ISO approved or de facto standards exist in these developing areas. The CBP could possibly evolve to include such facilities.

9.3 Protocol Evolution

The JNT coloured books and the CR82 Ring protocols are not ISO standards nor are they likely to be. It will be necessary eventually to change the protocols on both WAN and LANs in the light of current development work on protocols to whatever emerge as international standards. This will be a major change for the entire network community and will not come quickly.

10. PORTABILITY

Fortran 77 and Pascal will allow PERQ CBP software to be moved to and from other non PERQ computers. However it is recognised that even when programs are written in Fortran 77 and Pascal much work often has to be done to move them because of the inbuilt operating system dependencies. By using 32 bit, virtual memory Unix as a de facto standard execution environment it should be much easier to move programs in Pascal and Fortran 77 from one CBP Unix system to another.

Portability is also one of the reasons for backing national and international standards generally, hence the use of the GKS graphics package. GKS will be available on all SERC supported machines.

Portability of software is also one of the aims of the networking side of the CBP. Good communications are needed if software is to be easily shared by geographically dispersed research groups.

11. APPLICATIONS SPECIFIC SUPPORT

The CBP is expected to be expanded to include some items related to specific applications. These might possibly be the NAG library, RAL graphics library etc as well as software development tools from the STI, IKBS etc. In addition much applications specific software will be generated 'on top of' the CBP and which will be generally available but which will not actually be part of the CBP. The CBP is supposed to form the 'base' not the totality of available software.

12. GENERAL POINTS

a. Great stress should be laid on the fact that the CBP does not see single user systems as standalone systems. Networking is the key to file backup, mail, software update and interchange.

b. CBP links people just as much as computers.

c. CBP aims to back international standards if possible.

d. Software sharing and portability only really come when both the programming language and execution environment (ie operating system) are defined. The corollary is "it's OK to change the machine - just don't change the (user/program and program/operating system) interfaces".

13. REFERENCES

1. ISO Standard Pascal (BS 6192)

2. Ansi Standard Fortran 77

3. GKS draft ISO standard

4. UNIX Manual

5. PERQ glossy

6. PERQ hardware manual

7. CR82 UK Ring hardware specification
   CR82 Interface Specifications
   Orange Book

8. SERCnet X25 specification

9. TS29: Green Book

10. FTP80: Blue Book

11. JTMP: Red Book

12. MAIL: Grey Book

13. Transport Service: Yellow Book

14. TSBSP, BBP
    CR82 Protocol Specifications: Orange Book

15. Mixed Language Working
    A Williams
    RAL

APPENDIX A

THE PERQ

The Perq is a high powered, single user computer system with a high precision display system which provides a significant improvement in the quality and speed of interaction. Its main features are:

(1) High Speed Processor

Approximately 1 million 'high level' machine instructions per second giving around two-thirds the CPU power of a VAX 11/780. The CPU is micro-programmable for further speed gains.

(2) High Quality Display

A4 size, 1024 x 768 pixel, high resolution black and white display featuring 60Hz non-interlaced refresh rate which enables pictures to be moved cleanly and rapidly as well as giving a significant improvement in the clarity of text and diagrams equal to a printed A4 page.

(3) User Friendly I/O Devices

A 2-D tablet and voice synthesiser, allied to the high quality screen, enable a much improved man-machine interface to be created.

(4) Large Virtual Memory

A 32 bit address paged virtual memory system.

(5) Local Filestore

A 24 Mbyte Winchester disk and 1 Mbyte floppy give a single user a large amount of local storage capacity.

(6) Fast Communications

Local communication at 10 Mbits/sec via Cambridge Ring. Standard RS232 serial and IEEE 488 parallel interfaces are also provided.

A high quality, superbly interactive computing system is created if each investigator has his own single user PERQ linked to his colleagues' PERQs and other departmental computing resources by a Cambridge Ring, with inter-university cooperation being fostered by the National X25 network connections.