Managing system creation

D.K. Hitchins, M.Sc., C.Eng., F.I.E.E., M.R.Ae.S., A.M.B.I.M.

Indexing terms: *Engineering administration and management, Project and production engineering, Management*

Abstract: The paper introduces systems methodologies to illustrate the span of systems engineering activities and the system management structure appropriate to creating a complex and enduring system. Information/decision/action (IDA) systems provide a vehicle to demonstrate a variety of methodologies in operation. This class of system includes individual humans, companies, governments, computer-based data systems and, paradoxically, the system design/management team intent on creating the respective systems themselves. Systems are viewed as having seven ages: conception, design, development, implementation, transition utility and senility (which is followed by replacement). System creation invokes five subsystems: the operational facility, two development subsystems concerned with test-and-integration and incompany support, and two deliverable inservice subsystems for user/operator training and through life maintenance. System creation team structure is shown to comprise operations analysis, requirements analysis, system design, equipment engineering, software engineering, test and integration, and installation and commissioning; the systems approach emphasises the first three of these in particular, to reduce the generation of *ab initio* design errors. Functional decomposition and design option tradeoffs are demonstrated by example, and the establishment of end-to-end system dynamics is introduced using the ISO open system interconnection philosophy. System management organisations are outlines, and the allocation of resources by time within the project team structure is explored pictorially. Transition into operational use is highlighted as an area of special concern and, finally, the key issues of the systems management approach are identified.

Introduction

Much has been published on the subject of systems engineering, its methodologies, techniques and benefits. Relatively little has been produced on the management of systems engineering projects; this paper seeks to remedy that shortfall.

The word 'systems' is so widely used that it has become necessary to provide a taxonomy. The broad class of systems considered in the paper is that in which humans or groups of humans gather information, reach agreements and make decisions based on that information and implement consequent actions. This information/decision/action (IDA) classification has been selected not only because it is so wide in application, but also because it contains probably the most difficult categories of system design management, those concerned with multi-user/machine interactions and human factors.

The paper may be read at several levels. For example, it may be viewed as an embryonic company procedure document, or a largely pictorial description of IDA system creation. On the other hand, the principles and graphical techniques employed in the paper can be viewed as simple examples which might be re-employed in a much wider context, from detailed analysis of production organisations to the broad design of management systems.

In a similar vein, the paper could be viewed as arguing the case for systems engineering as a separate discipline, and in a way it does. Nobody questions the role of the architect in civil engineering, although his individual engineering skills are no different from, and may be no better developed than, those of the civil engineering contractors. His major contributions are creativity, structure and balance within constraints imposed by a particular situation. And so it should be for the system designer; he too covers a broad span of disciplines and seeks to bring creativity, structure and balance to design. And he too deserves separate recognition.

Paper 4783A (M3, M4), first received 14th November 1985 and in revised form 9th January 1986

The author was formerly Technical Director of Racal Defence Systems Ltd, Richmond Court, 309 Fleet Road, Fleet, Hampshire GU13 8BU, and resides at Sarum House, 4 Clevedon Court, Frimley, Surrey GU16 5YW, United Kingdom

The paper is presented in 'storyboard'. Each page of text is accompanied by a diagram, which contributes directly and without separate explanation to the topic under discussion. A suitable procedure is to read the first section of the text, examine the diagram and then return to the remaining text for expansion.

1 Information/decision/action (IDA) systems

This paper is concerned with the creation of the wide class of IDA systems which share the characteristic that they all involve human decision making. System creation to support this decision process must reflect the complex nature of the human relationships between the system user and operators. Effective management of system creation is characterised by its own skills and techniques.

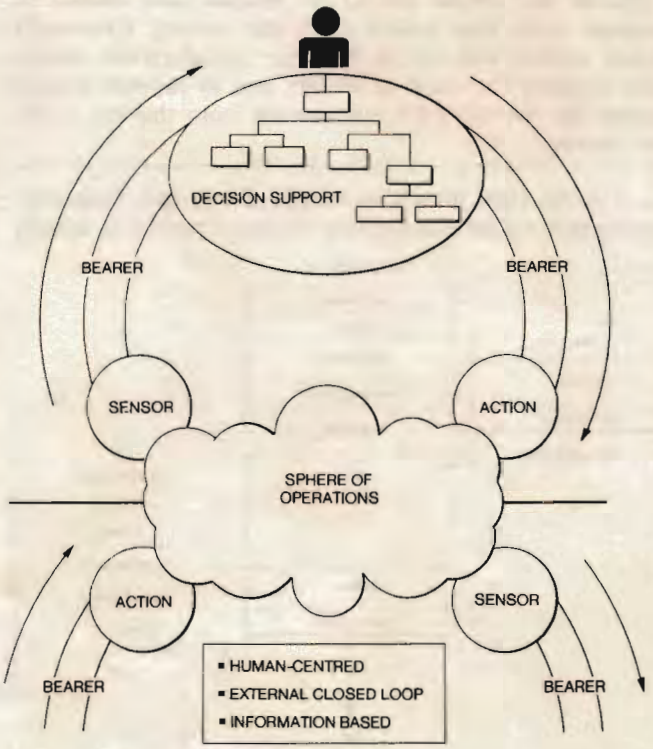

Fig. 1 *Information/decision/action systems*

Fig. 1 shows the information/decision/action class of system with which this volume is principally concerned. It comprises sensing, communication, assessment of the sensed information, human-based decision and consequent controlled action.

The class of system embraces a very wide variety of members, including:

(i) individuals and groups of humans
(ii) piloted vehicles
(iii) companies
(iv) governments
(v) air traffic control
(vi) finance and banking
(vii) management information
(viii) command and control information
(ix) intelligence
(x) IDA system management of creation.

Excluded from the class, by virtue of their fully automatic nature, are process control systems not requiring human decision. However, process control may occur in the subsystems from which information/decision/action (IDA) systems are constructed, particularly in sensing and execution subsystems.

IDA systems are generally highly complex. Complexity arises principally from the intimate relationship which must be established between the IDA systems's components and the human organisation which it is intended to serve. The decision support subsystem in particular can comprise many operators, each with his own formation handling facilities, concerned with different aspects of an overall decision process. The 'real' decision process is occurring, then, between the system operators, while the system in the Figure must provide carefully tailored support to each individual in the decision chain. This paper is concerned with the means for creating that synergy between operator and system in a cost-effective way, using the system management approach.

2 System lifespan

A system will evolve during its lifespan and should be designed with that ability from the outset. Eventually system senility will set in, however; good system design seeks to delay the onset of senility and to support ease of transfer for the collected information from the old to the new system.

Fig. 2 shows eight phases in the life of a system, from conception to eventual replacement. System creation is usually a shorter period than system operation. The creation process must, then, recognise from the outset that the system will evolve during its useful life, and 'design for evolution' is essential.

Transition is the period of introduction of the system into operation. Designers often envisage a system only as they expect it to be during its operational state of utility. Achieving this goal requires special facilities and procedures which may be peculiar to the transition phase, so that the human operators can transfer smoothly and with enthusiasm from the old to the new system.

An objective of system design should be to stay the onset of system senility, the causes of which will be discussed later. The onset of senility is to some extent, however, a self-fulfilling prophecy. Conception, design and development of a new system may be so protracted that senility must be anticipated, and the new system may be ready before the old really needs replacement.

An appreciation of the terminal phases of a system's lifespan is valuable in the system creation process. As the figure shows, the new system will interface with the system it replaces and information will be transferred to the new system during start-up. It is, therefore, sound practice in that context to design for ease of replacement, so that information collected and stored, perhaps at great expense, in the old system can be transferred to, and retained in, the new.

3 System life cycle

System designs contain the seeds of their own eventual decline. The management of systems creation generates system options and balances their merits against life-cycle cost so as to maximise the period of cost-effective system utility.

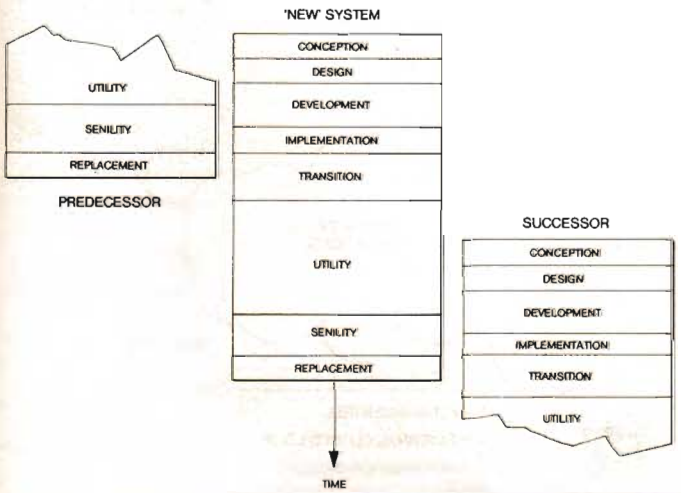

Fig. 2 *System lifespan*

Fig. 3 consists of a graph with axes of cost per unit time and effectiveness. The progress of a typical system is plotted against these axes. Rising costs, with no capability, start the cycle for development and implementation. During transition the system starts to become effective and the rate of expenditure may fall. The period of utility is marked by rising effectiveness and, in mid term, rising cost of ownership. Finally, effectiveness starts to tail off with costs still rising into senility and ultimate replacement.

The Figure also shows some of the causes of the cost-effectiveness cycle. Designs are usually frozen early in development, and it is common for hardware in particular to be dated even before entry into use. Protracted development and implementation also result in a mismatch between the evolving user requirement and the more static

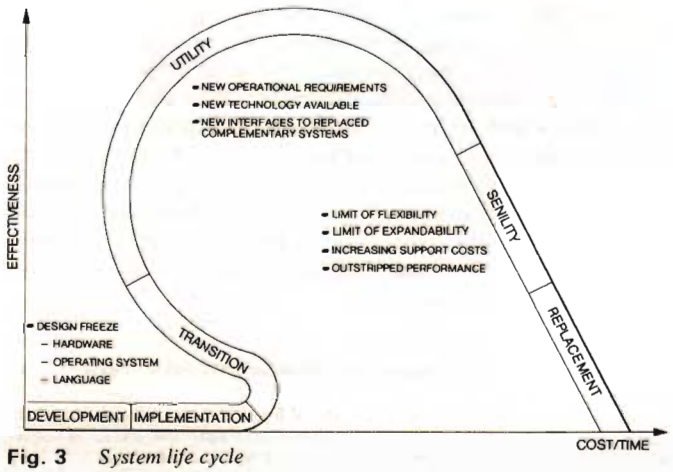

Fig. 3 *System life cycle*

design, so that design changes can be needed during transition as well as the period of utility. The new system will also interface, in the wider system context, with complementary systems which are at different phases in their life cycles, and new replacements will arise as situations change.

These changing requirements can be accommodated only by providing inherent and accessible flexibility and spare capacity in the system design. Even these pledges to adaptability must all be redeemed eventually, at which time the system can no longer evolve in step with the changing wider system and must eventually decline.

The system design approach recognises these evolutionary factors and trades off performance and continued performance, on the one hand, against life cycle cost, on the other. The generation of system options and the tradeoff process are a fundamental responsibility of systems management.

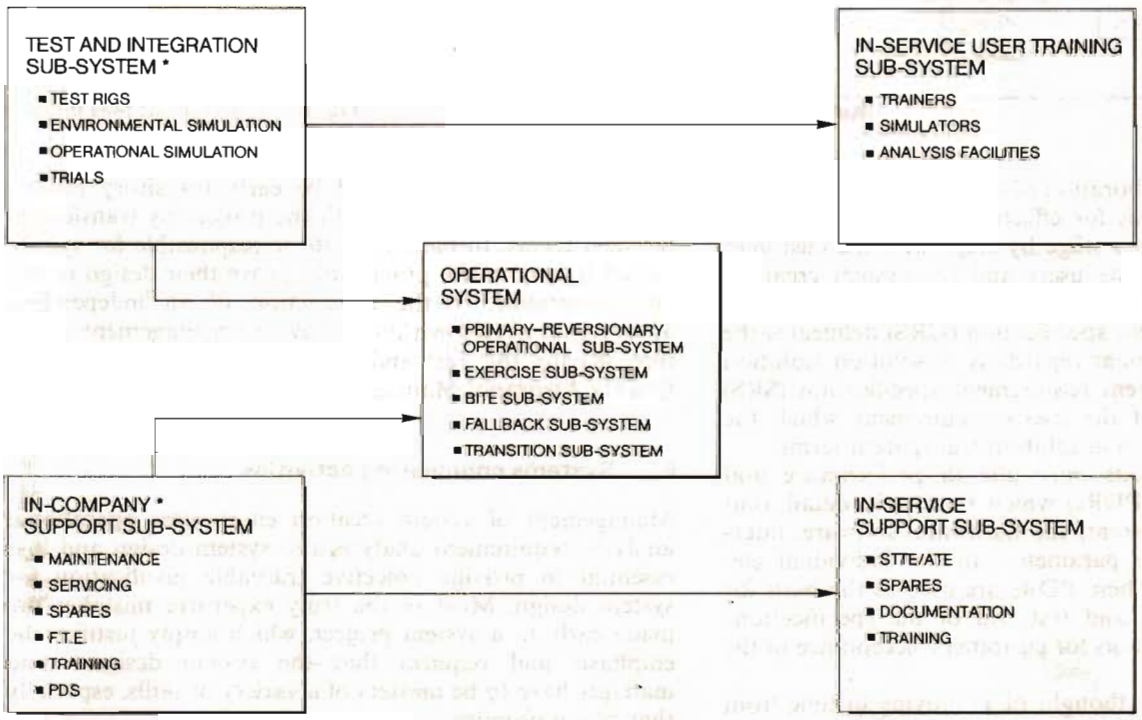

Fig. 4 *Project subsystems*

4 Project subsystems

Each system is comprised of at least five related subsystems, and the operational system should contain its own reversionary, fallback and transitional capabilities. Inservice and incompany subsystem compatibility can be promoted to maximise life-cycle cost effectiveness.

Systems and their eventual users need to be supported. Consideration of a system project shows that it comprises at least four major subsystems in addition to the operational system required by the customer. These four are:

(*a*) The Test and Integration Subsystem needed to prove the developing operational system's elements

(*b*) The Incompany Support Subsystem needed to maintain both the test and integration facilities and the developing operational equipments

(*c*) The Inservice Support Subsystem which provides spares, test facilities, training, documentation and development facilities for the user's operational and inservice training systems

(*d*) The Inservice User Training System (to train operational users) comprising trainers, simulators and analysis elements, and usually separate from the operational system *per se*.

Of special note is the composition of the operational system; it should comprise the five integral subsystem as shown in the Figure, so that it contains its own reversion (secondary system to operate when the primary system fails) and its own transition features, as well as the ability to detect faults and exercise operators.

A design which is intended to be life-cycle cost effective will capitalise on the potential similarities between the inservice and the incompany support systems and between the user training and the test and integration subsystems. For example, the customer's need for automatic test facilities in service can influence the choice of incompany test facilities, so that test access, test techniques, test software and test hardware are compatible. This approach offers shorter timescales and reduced costs, as well as more harmonious transition and eventual contractor support.

5 System project team structure

System creation comprises seven phases which can be reflected in the team organisation. The resultant teams are bound to each other through the formal use of specifications which are developed, agreed and maintained in cooperation with the customer. The team management is ideally established so that system design and development must be proven to the satisfaction of an independent test authority within the project, supported by the QA authority.

The system creation process comprises seven major phases. Project organisation provides seven equivalent functional groups. These tightly bound functional groups provide discrete outputs as shown in Fig. 5. Coupling between the groups is provided by formal specifications for each of the five subsystems shown in the previous topic.

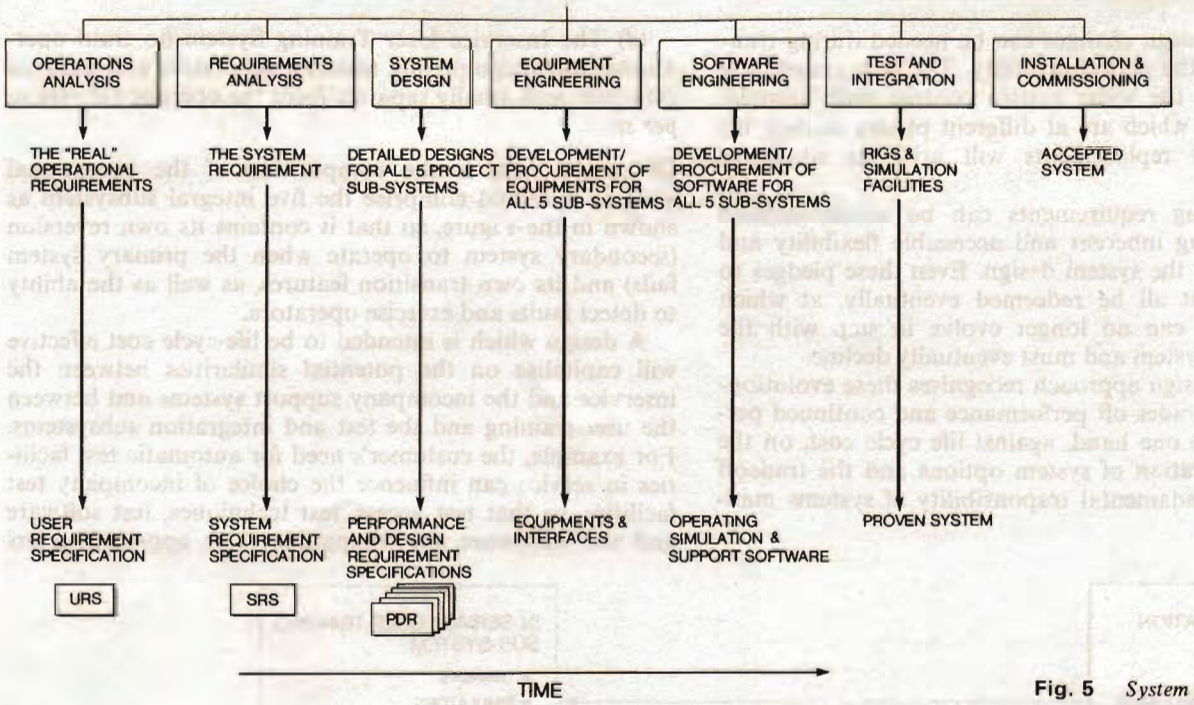

Fig. 5 *System project team structure*

The progressive elaboration of specifications is an essential if unromantic basis for effective system creation; the specifications are agreed stage by stage with the customer and evolved as both the users and the system creator's ideas develop.

The user requirement specification (URS) delineates the user's overall requirement regardless of solution (solution transparent). The system requirement specification (SRS) defines those parts of the user's requirement which the system will satisfy, again in solution-transparent terms.

System design results *inter alia* in performance and design requirements (PDRs) which specify, in detail, (but again solution transparent) the hardware, software, interface and performance parameters of the individual elements of the system. These PDRs are used as the basis for bidding, development and test. All of the specifications come together as the basis for customers' acceptance of the system.

The project may be thought of as moving in time from left to right on the Figure. Thus, operations and systems requirements analysis should be early transitory phases. Personnel may also move with the project by transferring between teams. In particular, those responsible for system design should, where practicable, prove their design in test and integration to the satisfaction of an independent authority provided within the overall management structure, usually the Test and Integration Manager with the Quality Assurance Manager.

6 Systems engineering activities

Management of system creation emphasises operational analysis, requirement analysis and system design and it is essential to provide objective traceable justification for system design. Most of the truly expensive mistakes are made early in a system project, which amply justifies the emphasis and requires that the system designer and manager have to be masters of a variety of skills, especially that of compromise.

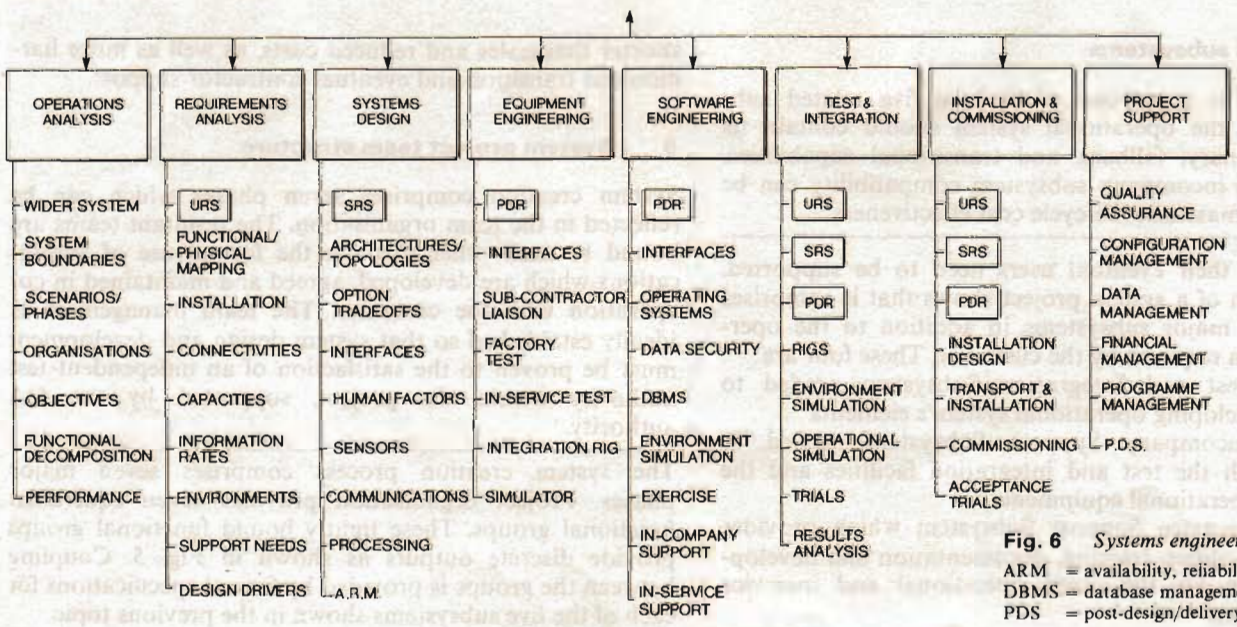

Fig. 6 *Systems engineering activities*

ARM = availability, reliability, maintainability
DBMS = database management system
PDS = post-design/delivery services

Fig. 6 shows a complete systems project structure with project support added to the seven chronological functions of the previous topic. Project support exists throughout the project life.

Operations analysis, requirement analysis and systems design can be seen as containing 'wider system' activities, and emphasis on these three areas is characteristic of effective systems creation. Following topics will highlight management methodologies in these areas particularly.

The activities shown under the various headings are generic as far as is practicable. However, the application of IDA systems requires highlighting of the human factors and information handling areas in particular. The system designer, in particular, has to master a variety of trades including:

(i) architecture analysis
(ii) computer science
(iii) control theory
(iv) human factors
(v) software structures
(vi) logistics and maintenance.

His principal skill must be in none of these areas, however; it must be in the ability to weigh and balance the many factors impinging on, and influencing, a design. The good system designer will not leap at the first solution he comes across. Instead he will:

(*a*) generate optional system designs which meet the system requirement

(*b*) compare each of the options against the required design characteristics

(*c*) select and justify the preferred solution.

Most of the major mistakes in system design are made at the start, hence the system management emphasis on operational analysis, requirements analysis and system design.

7 System tasks and skills

The work to be done by the systems functional teams can be logically 'decomposed' in stages. As an example, Fig. 7A shows stage one for operations/requirements analysis and systems design and Figs. 7B and 7C show stage two, the beginning of formal work breakdown, which highlights the system skills associated with each particular system.

The activity areas indicated in the preceding Section must be systematically structured so that the creation process is controlled and balanced. One technique for developing a logical workflow is the R-Net (requirement network) publicised (but not fully documented) by Robert Lano of TRW in a series of lectures he gave for the Technical Marketing Society of America in 1979, in London. Fig. 7A shows a high-level R-Net for operations and requirements analysis combined, and for system design. The R-Net is entered at the top and branches marked '&' must all be pursued, while branches marked '+' are alternatives; there are no '+' branches in these examples. The R-Net is much simpler than, say, a PERT Chart; items in parallel between two vertical &s can be executed together, serially or in any order. The Figure is concerned simply with their logical relationship.

Figs. 7B and 7C are also example R-Nets, but at the next level of detail. They too address operations/requirements analysis and system design, and are topographically mapped identically to the first Figure. Each block in the second-level Figures is marked with a work breakdown structure reference number which refers to a full description of the work to be done, skills required, resources required, outputs etc.

Examination of the R-Nets shows not only the tasks to be carried out, but also presents the skills which are particular to the system design process. The Systems Creation Manager is responsible for promoting and encouraging a dynamic, creative, even lateral thinking, environment during the early system design phases so that a full and balanced set of design options can be generated, reduced, massaged and detailed. Thereafter, the continuing gradual evolution of the preferred system design will require that the original concepts and tradeoffs be traceable, which, in turn, means that system design will persist at some level throughout the project's life.

Following Sections will highlight areas and methodologies of special relevance to system design.

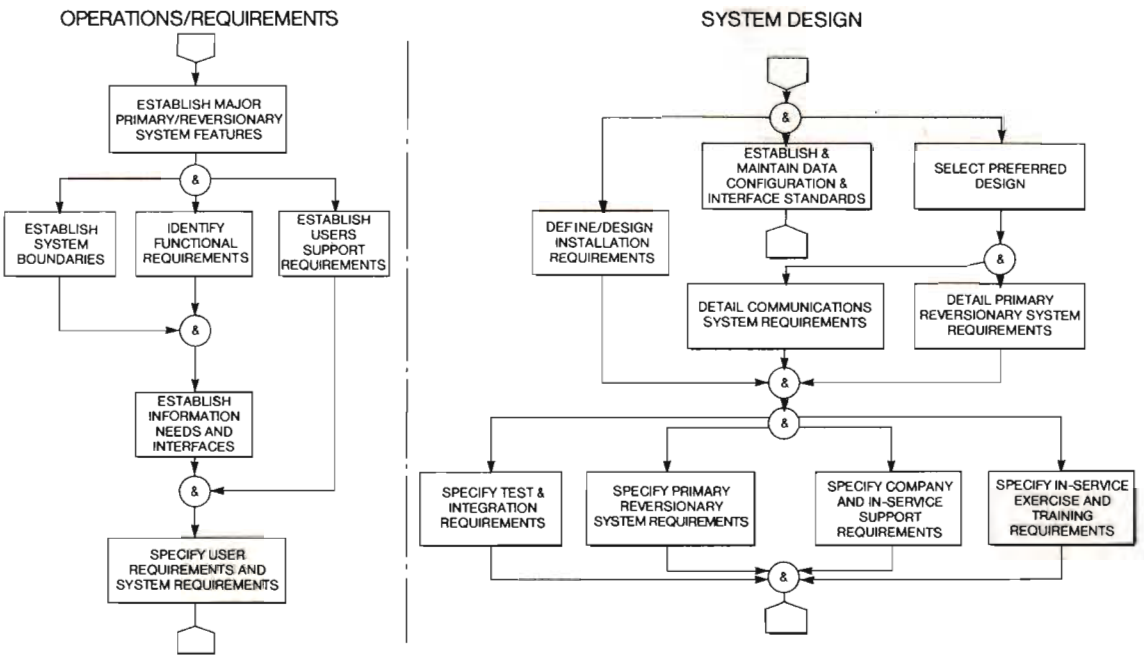

Fig. 7A *Tasks and skills: first level*

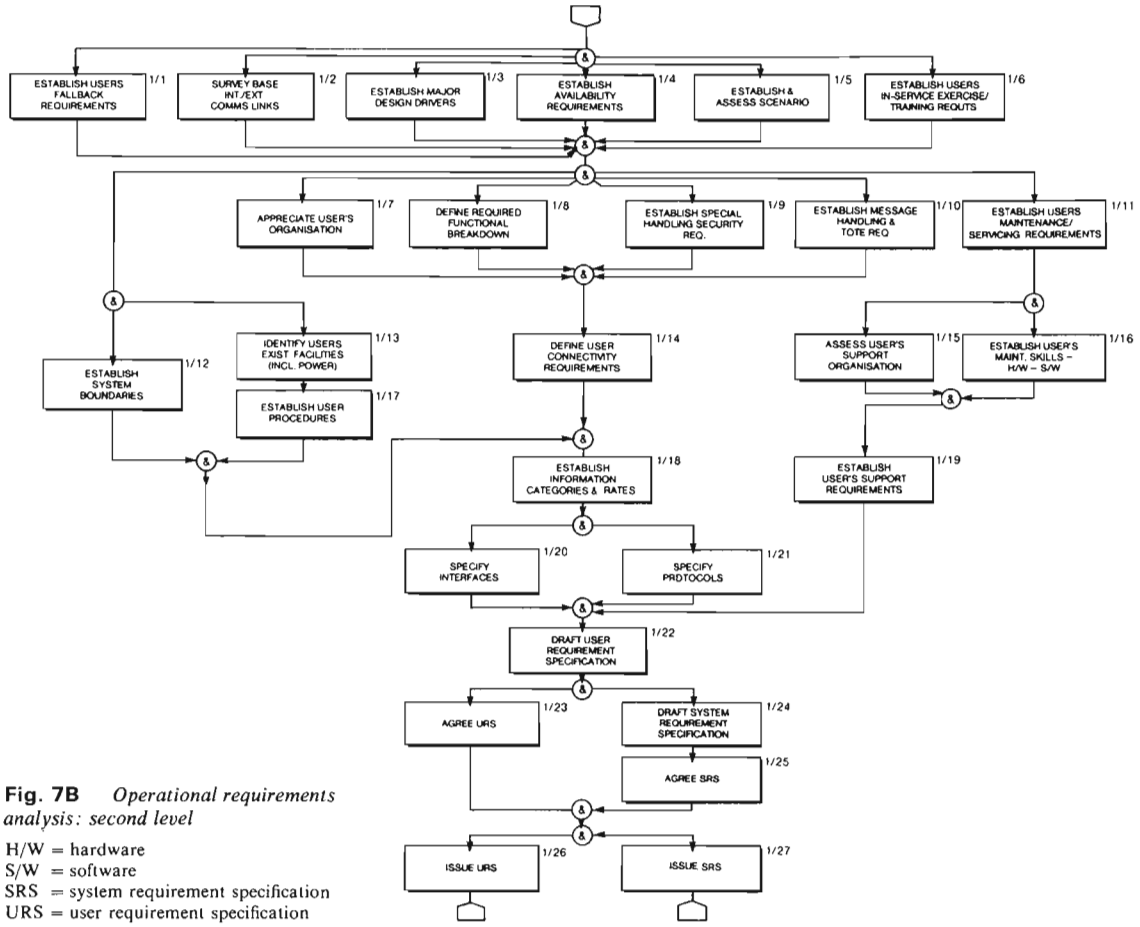

Fig. 7B *Operational requirements analysis: second level*

H/W = hardware
S/W = software
SRS = system requirement specification
URS = user requirement specification

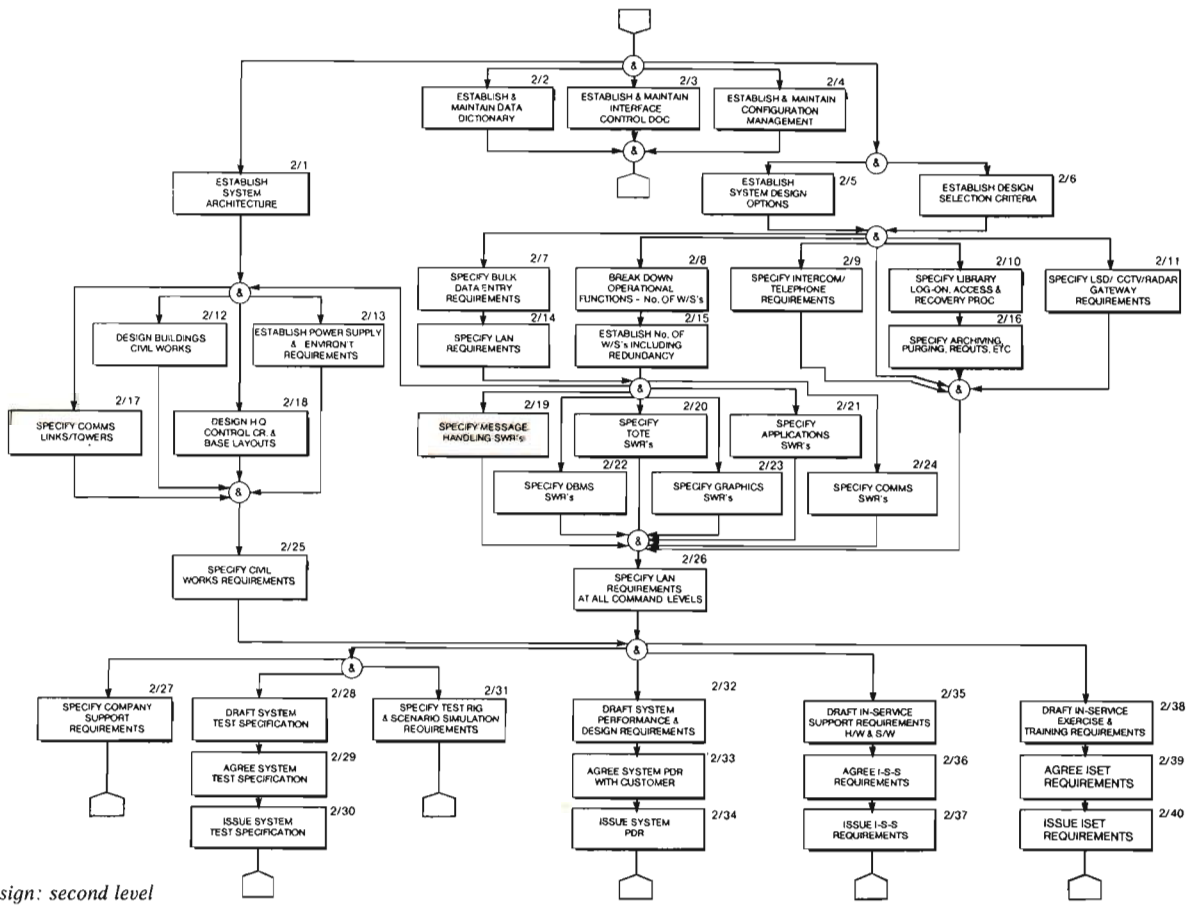

Fig. 7C *System design: second level*

HQ = headquarters
CCTV = closed-circuit television
DBMS = database management system
ISET = inservice exercise/training
ISS = inservice support
LAN = local-area network
LSD = large-screen display
PDR = performance & design requirement
SRS = system requirement specification
H/W = hardware
S/W = software
W/S = work station

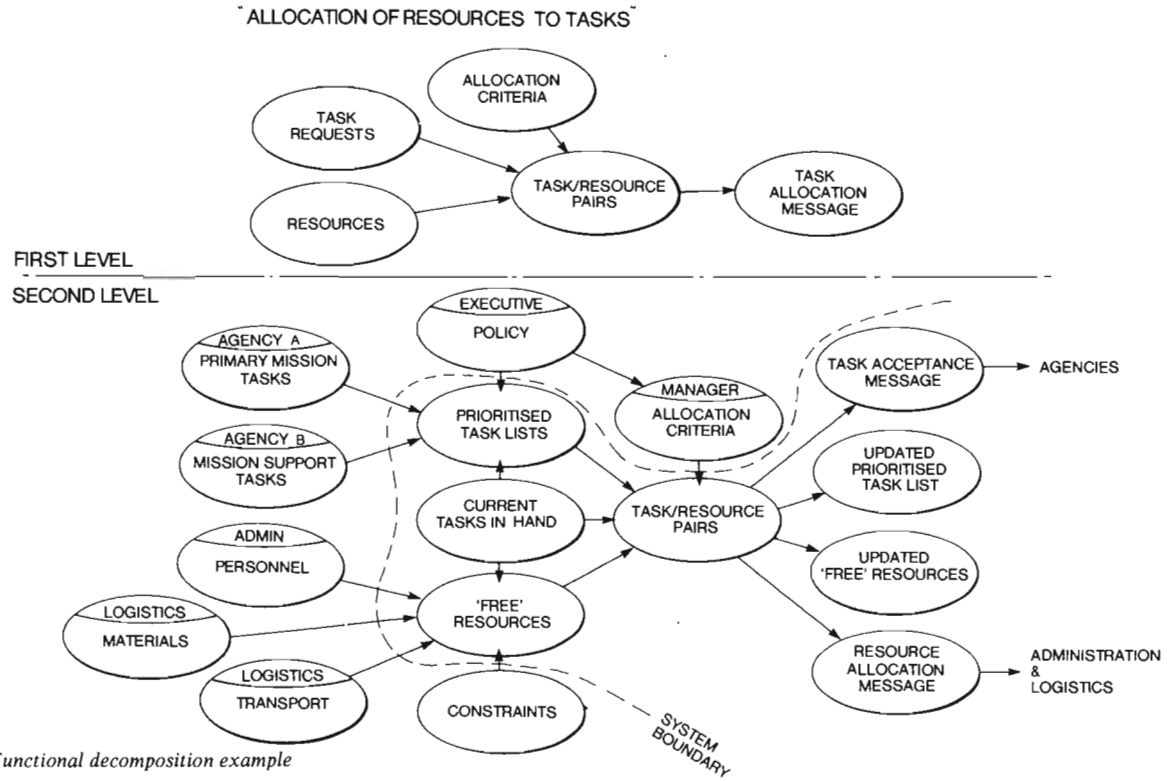

Fig. 8 *Functional decomposition example*

8 Functional decomposition

A common language is essential for effective dialogue between end-user and analyst/designer who must analyse the primary, reversionary and fallback systems on the same basis to ensure compatibility, and in a way that the end-user finds simple to understand.

Functional decomposition, part of operations and requirements analysis, is the systematic, progressive detailing of a function without regard to physical interfaces or other constraints. It is essentially a top-down creative process, which should proceed hand in hand with the potential end-user because it describes either his current tasks or future wishes.

As functional decomposition brings together end-user and analyst, a common language is needed which is at once easy to follow and quite specific.

The example shown in Fig. 8, data state design, presents data only in the bubbles and unspecified processes in the arrows. Successive detailing of a function, as shown for the simple case, 'allocation of resources to tasks', eventually leads to individual data pools which may represent, in some cases, database entities. However, it is important to realise that the IDA system requirement is comprised of only a part of each function; the dotted line separates out the system functions from the overall user requirement while keeping their relationship intact.

Fallback will probably require that each user function continues at some level, even when the end-system is 'down'; the data state design is also used as the basis for designing the reversionary (limited system performance) and fallback (no system performance) modes of operation so that these are compatible with the primary mode. Fallback in an IDA system generally consists of the operators working without part or all of the system. For fallback, they will need nonvolatile information during downtime. They will similarly need an alternative communication medium to receive and send volatile information. Fallback information system design may be layered to accommodate differing degrees and durations of main system failure.

9 Design drivers/option tradeoffs

Each IDA system is unique in realisation. Each design is influenced from the start of its creation by a variety of mutually inconsistent design criteria. An objective, visible and traceable means of comparing system-design options and balancing the many design influences is essential.

Each IDA system may have the same basic elements, but each is unique in realisation; the interfaces with the wider system and the characteristics of the particular requirement ensure that uniqueness. Classic systems approach requires the generation of objective measures of system excellence. Fig. 9 shows some of these principal design influences or design drivers for IDA systems.

Utility is concerned with performance of the primary, reversionary and fallback systems. Availability to the end-user is based on high system reliability and ease of maintenance. Adaptability, the staff of longevity for the system, is comprised of flexibility for evolution and expandability for growth. Interoperability is concerned with communication (in its true sense) between end-users in the wider system. Usability is the compatibility in design between the end-user in his working environment and the system. It includes the psychology of the end-user's perception of the wider system. Survivability for systems in harsh or military environments is a powerful design influence; it comprises avoidance of detection, self defence, when detected, and damage tolerance, when damage is sustained. Security is important for personal, financial, banking and military information systems.

Comparison of design options is a necessary feature of the systems approach. This is often undertaken using conventional weighting and scoring methods, which suffer from two principal deficiencies: the weightings are subjective, and the simple algebraic summation of weighted scores for different criteria is often invalid. The technique shown in Fig. 9 has been developed by the author to overcome these limitations; it is called rank matrix analysis (RMA). RMA ranks design options criterion by criterion and statistically analyses the resultant rank

Design drivers		Display system options (example only)					
		a 1 alpha	*b* 2 alphas	*c* 1 alpha + graphics	*d* 1 graphics	*e* 2 graphics	Row sums
Utility	Performance	5	4	2	3	1	15
	Fallback/recovery	$4\frac{1}{2}$	2	2	$4\frac{1}{2}$	2	15
Availability	Reliability	4	2	3	5	1	15
	Maintainability						
Adaptability	Flexibility	5	4	2	3	1	15
	Expandability						
Interoperability	Communications						
	Compatible protocols						
Usability	Human factors	5	3	4	2	1	15
	Man–machine interface	3	4	5	1	2	15
Survivability	Avoidance of detection						
	Self defence						
	Damage tolerance	$4\frac{1}{2}$	3	2	$4\frac{1}{2}$	1	15
Security	Data						
	Physical						
	Rank sum	31	22	20	23	9	105
	Preferred solution	5th	3rd	2nd	4th	1st	

Coefficient of concordance = 0.5102
Probability of random occurrence < 1%

Fig. 9 *Options and tradeoffs*

matrix for significant divergence from random (null hypothesis). Ranking is often the only objective comparison possible between options at the early stage of design. In the example, optional operator display configurations are compared and the pattern is shown as very unlikely to be random. Option *e*, the preferred option, is undoubtedly the most expensive, but cost effectiveness must be judged system-wide and over the full life cycle, not piecemeal, subsystem element by element. In practice, RMA and conventional weighting-and-scoring methods are often best used together, and their results compared.

10 Architectures and interfaces

System performance, survivability and resilience are founded on the basic architecture, which is comprised of nodes and links. Design of a flexible architecture is at the heart of effective system creation, but is often dismissed as obvious or unimportant. The most far-reaching consequences can accrue from adopting a poor architecture at the start of system creation.

System structure, architecture, topology, etc. are the basis for eventual performance in design but are astonishingly,

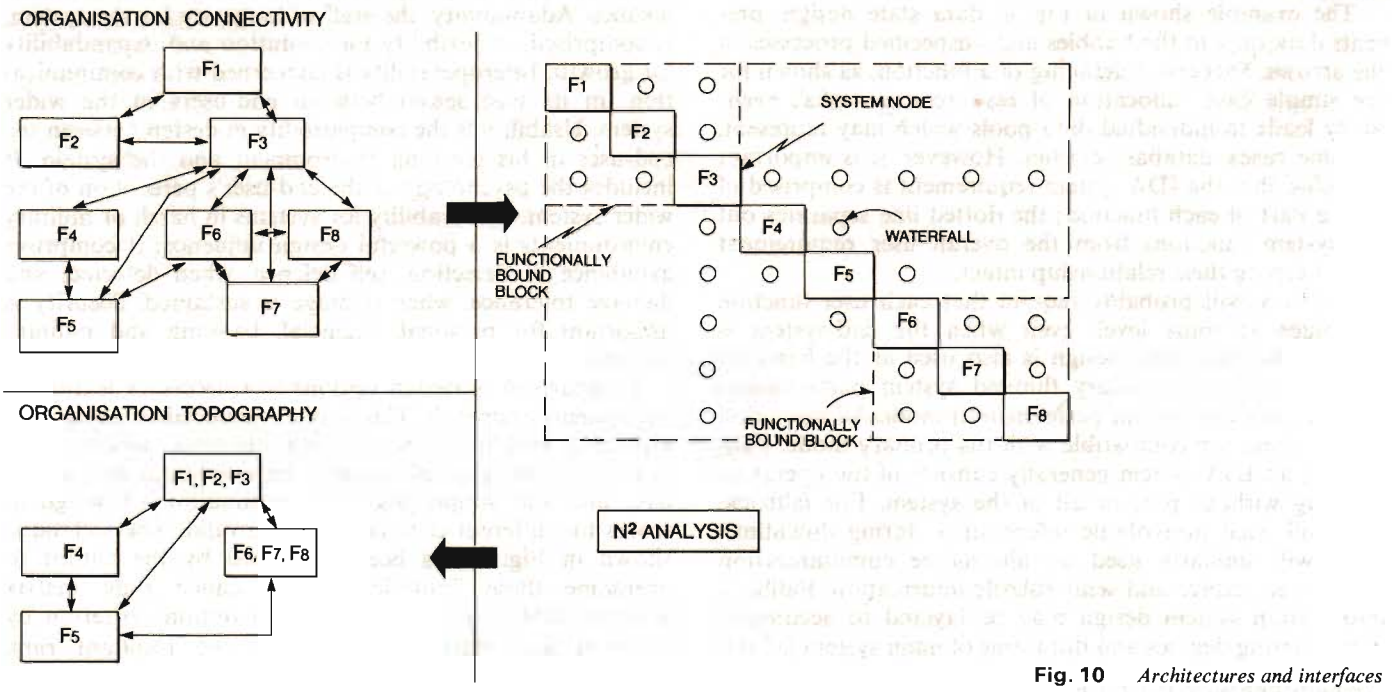

Fig. 10 *Architectures and interfaces*

often cursorily dismissed as 'obvious' or unimportant. Architecture is concerned with the identification and linking of nodes, nodal replication for survival and mobile operations, information replication for resilience, and the mapping of functionally bound subsystems on to physical architectures.

Top left of Fig. 11 shows a related set of entities. These could be people, organisations, software modules, distributed processors, The right-hand side of Fig. 11 shows the interfaces between these entities are represented in an N-squared chart [1] so-called because it comprises N rows by N columns giving N-squared rectangles. The leading diagonal represents the N entities, leaving N-squared-N interface rectangles. Each of these lies on the X-Y coordinates of two entities. Upper right rectangles represent 'down' interfaces in the hierarchy, while lower right rectangles represent 'up' interfaces.

The N-squared chart reveals three distinctive interface patterns in the organisation hierarchy. F_1, F_2 and F_3 have all mutual up and down interfaces represented; F_1, F_2 and F_3 comprise a tightly bound functional group, which would be a candidate for physical cohesion in design. F_6, F_7 and F_8 form a similar functional/physical group. F_3 is at the centre of an interface cross; it has interfaces to all entities and is a unique system node. Failure of F_3 isolates both halves of the organisation. F_3 is a candidate for reliability improvement, replication and/or protection. The third N-squared feature is the waterfall chain of command which shows orders 'rolling' down the organisation.

Finally, the topography at bottom left shows that the original hierarchy is essentially simple and highlights its underlying fundamental characteristics. Using the N-squared technique, particularly for large systems, permits the most complex architectures to be analysed, and their interfaces to be identified and controlled.

11 Interoperability

The advent of layered communication/information protocols promises to enhance performance and flexibility as well as interoperability. End-user compatibility may emerge as a new problem once the protocols are in place.

System design seeks to promote smooth continual flow. For production lines, the flow would be components modules and facilities contributing to the end product. For IDA systems information flow is the factor which transcends and unifies communication, processors, displays and operator interfaces.

Interoperability in IDA systems is essentially between human operators in the wider system community. A number of protocols is being developed which support this dialogue, the chief of these being (for IDA systems) the International Standards Organisation open-systems-interconnection (ISO OSI) seven-layer protocol shown in Fig. 11 [2]. This protocol consists of layers which buffer and isolate interoperability functions so that each layer interacts only with those immediately above and below it. The potential for evolution and growth is greatly enhanced by this 'damage limitation' layered protocol approach. The three lowest layers are concerned with the establishment of a networked communication facility. The transport layer provides an 'envelope' inside which information is sent over the network. 'Session' binds and unbinds groups of communicants. 'Presentation' formats, translates and presents information. 'Application' manages both the communication and information facilities in support of the end-user's particular activity.

As Fig. 11 shows, there are several layers needed above the seven-layer protocol to ensure that the human operators truly understand each other. The major obstacle to interoperability is not inability to understand — that can be detected and remedied; the principal risk arises from a mistaken belief that full understanding exists and that there is, therefore, no cause for concern.

It is not generally realised that the OSI concept can be very widely applied; it is not restricted to fixed-site computer-based transaction-processing systems but can readily extend to tactical, mobile real-time systems too.

12 Test integration and simulation

Test and integration require that the developing system be immersed in an environment to represent the extremes that might be met in practice by the end system, so that all system algorithms, capacities and performance can be established, tuned and proved. Simulation of scenarios, of nonavailable end-system components, and of system response will all be needed.

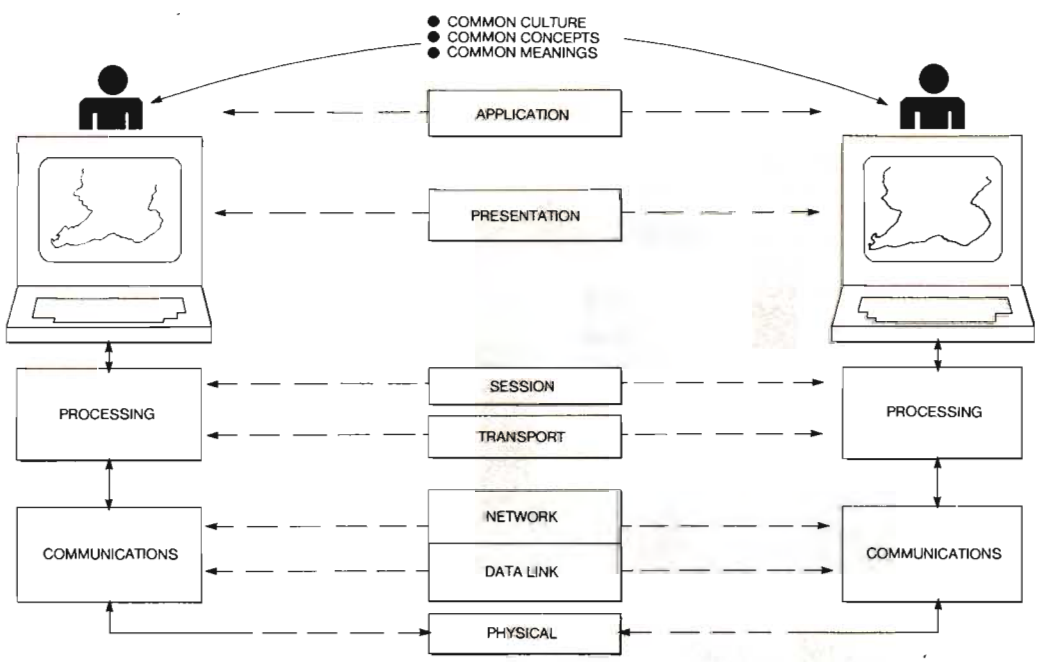

Fig. 11 *Interoperability*

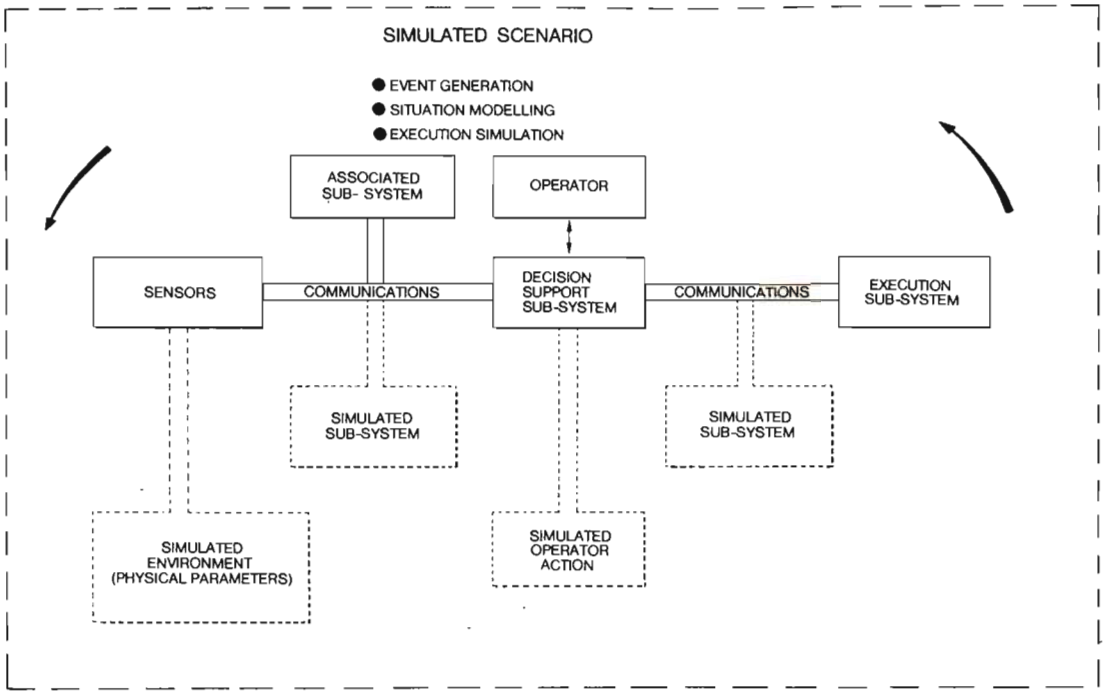

Fig. 12 *Test, integration and simulation*

The process of system creation will bring together hardware and software for test and integration. In any system of significance, the full testing of the system, man–machine interface, communications, software algorithms, etc. will necessitate the construction of a rig to represent parts of and interfaces to the wider system.

Progressive rig proving will require equipments and facilities to be interconnected, stimulated and analysed. There are three distinct elements to this process. Some equipments may not be easily obtained and their behaviour must be simulated. External events which activate system functions must also be artificially represented (e.g. radar returns to represent a situation which might only be met under extreme operational conditions). Finally, the response of the end-system which the rig represents must be simulated (e.g. the response of the eventual operator to the radar returns may change vehicle orientation and, hence, instruments and displays on the rig must be made to change too).

The process of simulating the local and wider environment surrounding the system under test can necessitate a set of simulation facilities which is more elegant in its way than the end-system itself. The development of naval and air real-time systems, in particular, requires elaborate simulation, which includes natural and man-made interference, garble and loss of information, simulated defects and the combination of situations and conditions most likely to 'break' the system.

13 System management organisation

Preceding Sections concentrated on the systems engineering aspects of system creation; the mangement of systems engineering is vitally important too, and will be highlighted in following Sections.

Unlike product companies which are physically partitioned in support of production, systems companies may be functionally partitioned around a project base. Progressive development of a systems company may lead through matrix management to the formation of a consultant's group.

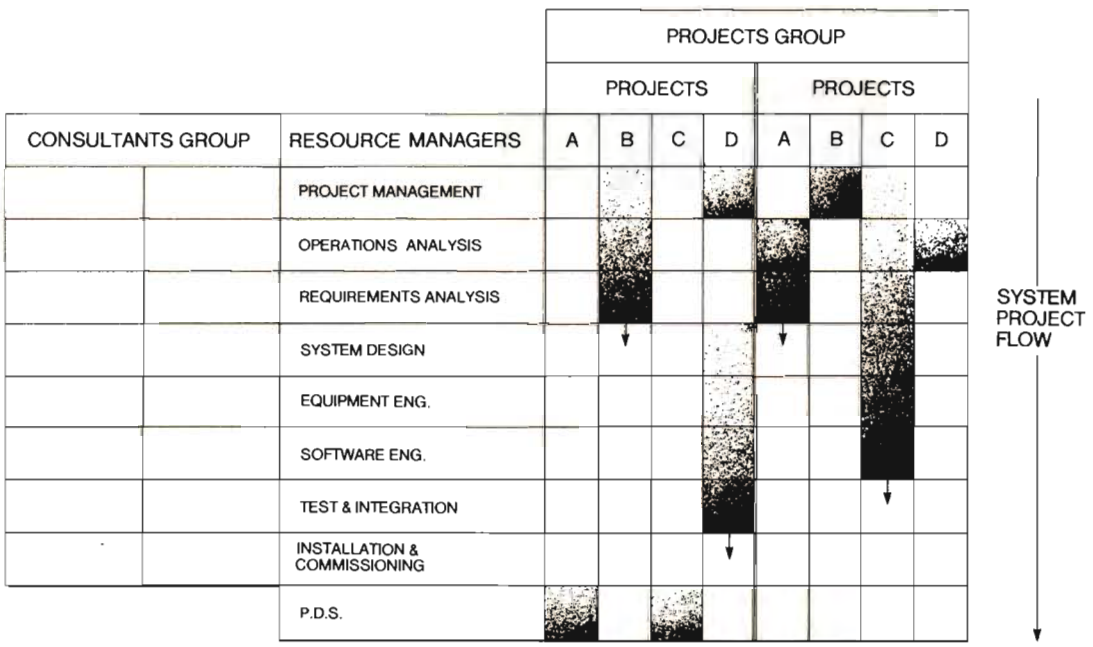

Fig. 13 *System management organisation*

Systems companies and product companies will generally be organised quite differently. For the product company, the production facility often represents a significant, and immobile investment in real estate and mechanical equipments. For the systems company, emphasis is on system-level design and software development, test and integration. As a result, product companies tend to be physically partitioned with goods inwards, stores, development laboratories, production areas, etc. Systems companies tend to be functionally partitioned around projects and groups of projects.

Fig. 13 shows a typical systems company organisation. On the right-hand side of the Figure a number of projects can be seen (*a*, *b*, *c*, etc.) at various stages of their progression through the company. The projects are grouped into two areas, probably according to customer category.

As systems companies grow, the projects mutually compete for resources. Fixed resources, such as the test and integration areas and software development facilities, may require separate managers. A matrix management organisation can grow as shown under the heading 'resource managers', with all project resources centrally managed.

As projects wax and wane some resources will temporarily become surplus. A good resource manager may 'sell' these surplus personnel resources of expertise as consultants outside the company, and the growth and eventual independence of a consultants group can be expected.

14 System project manning

System creation is presently labour-intensive, particularly in the areas of system design and software development. The various system creation activities tend to overlap in time, and it is to be expected that a considerable degree of work will continue during transition, post delivery.

The passage of a system creation project through a company has been presented, so far, as comprising sequential phases; in practice that may be an oversimplification.

Fig. 14 shows a typical set of manpower profiles for a systems project, with milestones along the top. The derivation of the requirement and the design can be seen, leading to equipment and, particularly, software engineering activity. The need for a design improvement is mooted, and it has consequences in equipment, software and test and integration areas. User training is shown during the development; end-users can usefully be incorporated into test and integration, both for their training and to smooth the path of customer acceptance of the system.

Noteworthy points include:

(*a*) the continuance of system design throughout the project

(*b*) the labour intensity of software development

(*c*) the overlap of the various functional activities

(*d*) the major requirements of the project postdelivery, particularly for inservice support facilities.

Attempts to cost major systems generally underestimate costs, often because the estimators believe that the system will be developed and completed according to a fixed schedule, and by a due date. In practice, this would be quite exceptional. The process of transition is not entirely within the control of the system-creation manager, who will nonetheless be responsible for tuning the system to meet the users' expectations and needs.

15 Transition

Transition is a greatly underrated phase of systems creation. Causes of slow or ineffective transition can be identified and ameliorated but essentially instant switchover from old to new system is an unreasonable expectation.

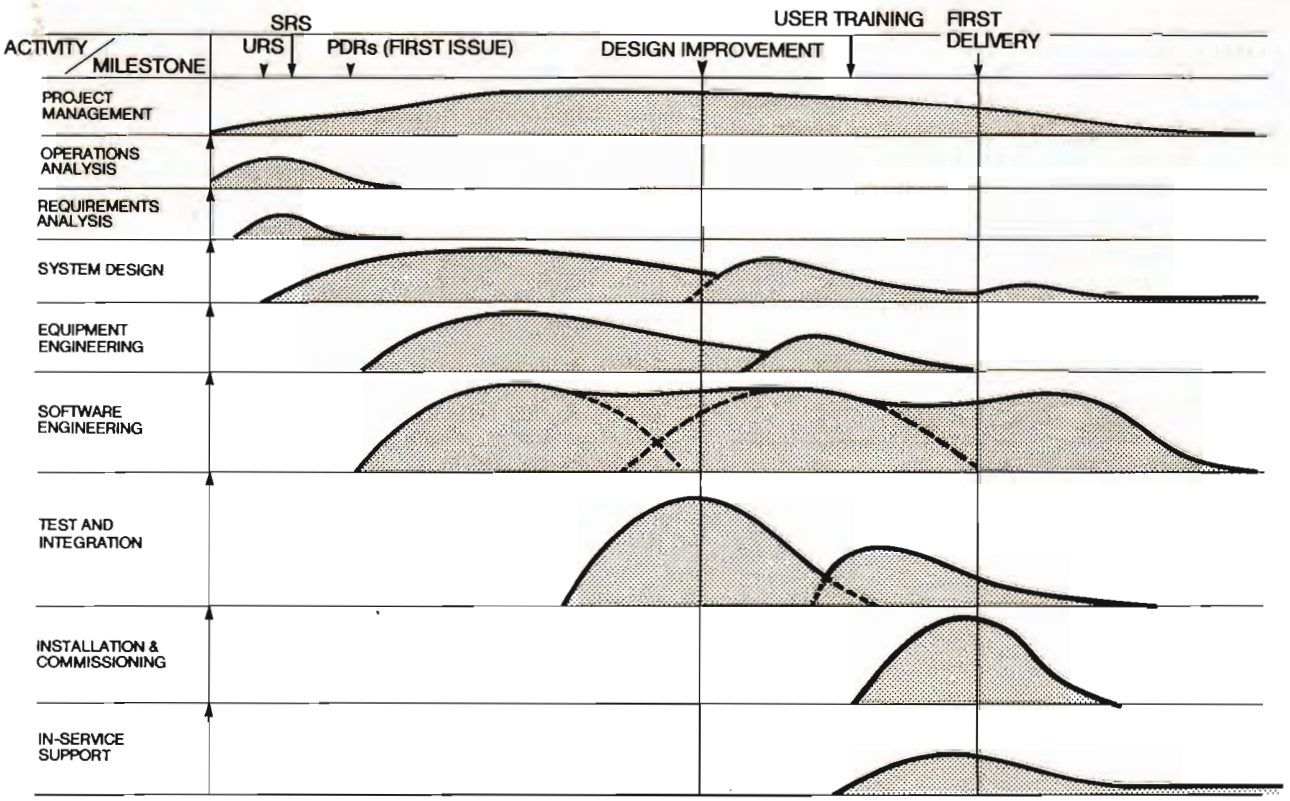

Fig. 14 *System project manning*

Transition factors	Probable cause(s)	Remedy
Slow transition	• Poor requirements analysis	• Discrete transition subsystem
Inoperative interfaces to 'wider' system	• Interface evolution	• Close control of interfaces through life
User resistance	• New system 'unattractive' • Advantages not evident	• User-friendly design • Co-operative implementation
Customer dissatisfaction	• Requirement has changed • False mental image	• Employ system flexibility • Early prototypes
Poor hardware reliability	• Infant mortalities/ • Poor design	• Design for reliability • Modular portable software
Poor software reliability	• Inadequate testing • Inadequate user trials	• Thorough tests & trials
Late support facilities	• Subordinate to primary system	• Emphasise cost of delay

Fig. 15 *Transition*

Transition has been highlighted in the preceding Sections; it is undoubtedly the most underrated phase of system creation. We expect to nurture a newborn infant, but a newborn system must be instantly capable.

Fig. 15 shows a number of reasons why instant capability will not occur. The potential remedies have a constant theme:

(*a*) continual attention to wider system interfaces
(*b*) attention to human factors at all levels
(*c*) co-operative user/creator design and evolution
(*d*) thorough test and trials prior to transition
(*e*) flexibility.

But, in essence, it is unreasonable to expect perfection. Instead, perhaps the systems creation process should indeed plan for a period of inservice nurture, with old and new systems running side by side and a gradual controlled transfer occurring from one to the other; such approaches have been tried with great success, but the desire for an instant 'switch' is hard to resist.

16 The systems management approach

The systems management approach is widely applicable, but is not yet widely applied. In some ways it is generated by an attitude of mind, but the benefits to be gained from this comprehensive, high-integrity approach require that it be considered as the separate discipline it truly is.

Wide applicability	- - - projects, companies, economics
Cost effective	- - - through-life optimisation
High integrity	- - - Methodical, comprehensive
Unique	- - - a discipline in its own right

Fig. 16 *The systems management approach*

The systems approach has almost universal applicability, but is not widely applied at present outside of the defence industry. Its applicability extends to development, production, administration, organisation and detailed design, although this volume has been limited principally to large IDA systems and management methodologies for large systems creation. The systems approach aims to be cost effective throughout the system life, but the pattern of expenditure differs from the bottom-up approach of selecting off-the-shelf, which is simple, tangible and generally unsatisfactory in realisation. It has also to be said that the systems approach can be misapplied, with generally expensive results; 'top-down' design can only be truly effective if it recognises the pragmatic limits of the 'bottom' towards which it is aiming.

Systems creation has to be based on integrity; it is a complex co-operative, systematic endeavour which requires continual customer involvement, plus a large element of forward thinking and planning to produce a satisfactory and comprehensive solution.

Lastly, the systems approach is different; it should be an amalgam of top-down and bottom-up; it designs backwards from senility to conception; it balances many inconsistent constraints; it designs for change and changes the design; it is concerned with architectures, survivability, topology and the wider system; it is, in fact, a discipline in its own right and is undoubtedly the way ahead as systems become more complex and intertwined.

References

1 LANO, R.J.: 'Operational concept formulation' (TRW Defence & Space Systems Group Publication, 1980)
2 KUHN, K.: 'The ISO/CCITT reference model for open systems interconnection and its military use'. 5th AFCEA European Symposium and Exposition, 1984, pp. 181–186