Controlling & Auditing Information System

Auditing System Developments

Prepared by:

Basuki Rahmad, Dr, CISA, CISM

2011

TABLE OF CONTENTS

1.	INTRODUCTION TO THE COURSE NOTES	1
2.	IT PROJECT FAILURES	5
3.	FEASIBILITY STUDY AND PROJECT INITIATION	12
4.	SPECIFYING USER REQUIREMENTS	30
5.	SYSTEM DESIGN AND DEVELOPMENT	46
6.	SYSTEM BUILDING AND ACCEPTANCE TESTING	64
7.	SYSTEM IMPLEMENTATION	72
8.	POST IMPLEMENTATION REVIEW	86
9.	RAPID APPLICATION DEVELOPMENT	89
10.	ALTERNATIVE DEVELOPMENT METHODOLOGIES	101
11.	THE EXTERNAL AUDITOR'S ROLE AND PEER REVIEW	105

1. Introduction to the course notes

These notes are intended to provide students with a summary of the various topics covered during the INTOSAI course on Auditing Developing Systems. The notes are not as a substitute for published text on computer auditing or on the many specialist skills that may need to be employed during a major IT project. Indeed, the notes identify a number of specialist areas, such as system design and development, where further training and research might be necessary to achieve audit objectives.

The following chapters cover the life-cycle of an computer system from strategic planning through to post implementation review. In the time available the course can only provide students with an overview of the main activities that take place during the life-cycle, but this will be sufficient to introduce the principles that organisations should apply to the planning and management of IT projects. A good knowledge of these principles will help the auditor to identify potential problems that could result in an unsuccessful or abandoned project.

1.1 Why do audit get involved in system development?

The computer auditor's overall objective, like everyone else involved, is to contribute to the success of every system project. The computer auditor is best qualified to do this by :-

- helping control the 'exposures' resulting from the project, and...
- giving management reasonable assurance that this has been done.

'Exposure is the product of the probability of an unwanted event occurring, and the cost to the business should that event occur. It can be controlled by reducing the probability of occurrence, or their potential impacts, or both. The auditor's role is therefore to:

- identify exposures;
- assess whether they are within acceptable limits;
- consider cost-effective ways of reducing excessive exposures; and....
- making recommendations to management, and to the Project Manager, that are both timely and achievable.

The two main types of exposure that arise from any system development project are:

> **security exposures** : stem from inadequate built-in automated controls designed to validate data processing; to restrict access to data, ensure a sound separation of roles and to monitor system use; and to provide appropriate business continuity;

project exposure : relate to the financial justification for the project, and include such impacts as late delivery, cost over-runs, failure to deliver the anticipated business benefits and premature obsolescence.

These exposures break down into a number of more detailed risks - these are summarised at Annex 1- and these are dealt with in more detail in the following chapters. The amount of work that will be required to gain assurance in each area will

depend on the nature of the application being developed and the system development methods used; if recognised methods are properly employed, it is far easier to gain assurance that best practice is being followed.

1.2 Types of audit : internal audit

In describing the audit activities that might take place during an IT project, these notes make no distinction between the roles of internal and external auditors. However, for the purposes of this course, internal audit is considered to be.....

> *an independent appraisal within an organisation that operates as a service to management by measuring and controlling the effectiveness of the internal control system.*

Internal audit's main role during the system development process will be to advise on the provision of internal controls within the system being developed. However, depending on the terms set out in their Charter, Internal Audit might also be involved at each stage of development to assess whether the organisation's system development life-cycle is being correctly applied and is likely to deliver an operational system that meets both business needs and its users' aspirations.

The following chapters contain examples of the sorts of questions that an internal auditor might need to consider at each stage of the life-cycle. These are based on the risks set out at Annex 1 below, and they might be used as a basis for the development of more detailed audit programmes.

1.3 Types of audit : external audit

For the purposes of this course, external audit is taken to mean.....

> *an independent examination of the financial accounts to determine whether they represent a true picture of an organisation's financial position.*

The external auditor's objectives are therefore different to those of the internal auditor. It will be unnecessary for the external auditor to audit every stage of an IT project, as to do so would contribute little or nothing to the certification of the organisation's accounts. But this does not mean that the external auditor does not have a role to play during the development process. An insecure system often results in a more time-consuming audit (due to the higher risk of undetected financial error and system misuse), while a design that does not include an adequate financial audit trail might well result in an unauditable system.

Annex 1 : system development life-cycle risks

Information systems strategy

(covered in the IT Methods module)

Objective : to obtain and implement information systems in a manner which maximises the ability of management to achieve the organisation's aims and objectives.

Risks are that the organisation fails to:

- establish a strategy which provides a statement of its objectives, scope, depth, and boundaries in terms of internal organisation and timescale;
- describe existing information systems and technology, identify available resources, and define known problems;
- define those factors that may limit the strategy scope
- establish the extent to which information systems will and will not be used to help provide the information required by the business;
- assess the costs and benefits of a chosen strategy;
- specify the framework of policies, standards and procedures to be adopted;
- provide an adequate planning, financial control and implementation infrastructure;
- devise and implement migration plans.

Annex 1 : system development life-cycle risks

Systems under development

Objective : to ensure that systems under development successfully meet the criteria of specified stages in a development life-cycle .

Risks include :-

- delivered systems do not meet user needs;
- delivered systems contain inadequate controls;
- systems function inefficiently
- failure to develop robust and secure systems;
- excessive future maintenance cost - systems cannot be maintained;
- incompatibility with other systems
- limited development/enhancement potential;
- poor quality product

Annex 1 : system development life-cycle risks

Procurement

Objective : to obtain goods to specified standards, at reasonable cost, within budget and by deadlines.

Risks are that :-

- goods do not meet user needs or requirements;
- goods are not available when required or supplier fails to deliver;
- goods are not of the required quality;
- goods are purchased without adequate authorisation;
- goods are purchased at inflated prices;
- government/legal procurement directives/legislation are not observed;
- inadequate Business Case;
- maintenance arrangements give poor value for money.

Annex 1 : system development life-cycle risks

Contracted-out services

Objective : to obtain services from a supplier in accordance with a specified set of terms governing requirements, obligations and remuneration.

Risks include :-

- failure to maintain service levels;
- monopolistic relations develop;
- contracts inadequate for the enforcement of supplier obligations - including confidentiality of client data;
- inadequate management of the contract;
- escalating cost.

2. IT project failures

2.1 Background

The remainder of this chapter places the course in context. It uses a number of case studies of unsuccessful UK government sector IT projects to illustrate why IT projects often take longer and cost far more to complete than was anticipated, and are sometimes abandoned altogether (often at considerable cost).

IT projects cover a wide range of scenarios, are often lengthy, and are rarely straightforward. At one extreme are IT projects that involve selecting and installing an "off-the-shelf" package (of which there are now hundreds of examples) onto an existing computer. Although such a task might seem be a straight forward, a surprising number of organisations continue to pay insufficient attention to specifying (and hence understanding!) their business requirements, training their staff to use the new system, and testing it thoroughly before release. As a result, the package often costs more and takes far longer to install than was anticipated, and sometimes fails to meet business needs.

At the other extreme are the large IT infrastructure projects where the object might be to design and build a computer system to administer national taxation or social security benefits. IT projects in this category often cost tens of millions of dollars and take years to complete. They also involve many specialist skills covering very demanding activities, such as business analysis and planning, systems analysis and design, project management, and various programming skills.

The main academic research studies that have been carried out find that IT projects commonly experience time over-runs, are often over budget, do not perform in the way expected, involve severe strain on the participants, or are cancelled prior to their completion with the waste of considerable sums of money. These problems relate essentially to size, complexity, long time-span/uncertainty, and management (even experienced project managers continue to face very challenging problems that they have never previously encountered). And especially in the government sector, external factors to the project, such as budget cuts and the need to meet politically imposed deadlines, add considerably to IT project risks. For these reasons, unsuccessful and abandoned IT projects are not uncommon.

In the UK NAO's experience IT project problems can be grouped under the four headings which are described below. But first, some case studies to illustrate the types of problems that often arise.

2.2 Case studies

Case 1: West Midlands Regional Health Authority

In 1989, the Authority contracted for the development of an electronic trading system. The project Business Case anticipated expenditure over the first five years of £5.3M. However, the Authority agreed with the developer to market the software to other potential users, and that the cost of development would be offset by anticipated royalties of £3.9M.

By 1993, the contractor had billed the Authority for £7.3M and, because of weaknesses in the contract covering licensing arrangements, the Authority had only

received royalties of £12,000. Furthermore, the system did not meet the Authority's needs and had to be abandoned.

Among the main concerns to arise from this project were:

- the Business Case was not viable because it contained information which was highly speculative and unrealistic;
- the Authority did not update the Business Case to take account of the changing cost/benefit profile for the project;
- the Authority did not monitor and control projects costs effectively.

Case 2: The London Ambulance Service

The London Ambulance Service "Computer Aided Despatch" system was developed at a cost of around £7.5M. It was implemented in October 1992, but operating problems with the system were experienced from the outset. On 4 November 1992, the system failed completely and this resulted in widespread (and highly publicised!) problems in controlling the London ambulance fleet and conveying patients to hospital.

The system was subsequently found to be unworkable. It was abandoned and the London Ambulance Service's Chairman and Chief Executive resigned.

Project problems included the following:

- the system was highly complex and contained much unproved leading edge technology
- development and implementation timetables were unrealistic;
- there was a very poor standard of project management, with no one person clearly in overall control;
- there was very little communications between the project management team and the organisation's management board, hence top management were unaware of the problems that were developing;
- the new system was introduced into a very poor climate of industrial relations, and ambulance staff resented it. The system therefore lacked "ownership";
- the development contract was placed with an inexperienced software house with no track record in the business, partly because.....
- too much emphasis was placed on contract cost to the detriment of quality.

Case 3 : Food and Agricultural Organisation, Rome

In 1985, the FAO placed a contract for US$6.6M for delivery of a new accounting and personnel system to be implemented by the end of 1988. The project was implemented in 1990 at a cost of US$30M.

Although the first phase of the project was completed successfully, the two later phases (mainly treasury, financial accounting and travel systems) resulted in widespread accounting and operational problems. By July 1991 over $400m lay uncleared in suspense accounts; there was a processing backlog of 25K items; and a ledger imbalance of US$35M. Remedial action cost US$4.9M including US$1M spent on consultants.

Key weaknesses included:

- the system was seriously undersized and could not provide a satisfactory response in relation to the workload;
- there was limited user involvement during development;
- key development procedures, standards and controls were either modified or dropped altogether due to resource constraints;
- there were various skills shortages, but particularly in project management.

Case 4 : Foreign and Commonwealth Office

In 1989, F&CO introduced a new computerised accounting system. It did not work properly; the accounts had a net imbalance of £5.3M and £26.4M was held in suspense.

Parallel running was delayed by supplier problems, but when undertaken the old and new systems could not be reconciled. Before this problem could be resolved the existing system, which comprised obsolete equipment, suffered a major hardware failure. It could not be repaired and there was no backup system. The F&CO were therefore obliged to transfer their work to the new system even though they were aware it contained errors and was not in an acceptable condition for live use.

The organisation's final accounts, which should have been certified in August, were not produced until November, and were derived from ledgers that did not balance. As a result, they could not be audited in full, and the audit certificate was heavily qualified. A significant amount of consultancy support was needed to re-establish full accountability within the organisation.

The overall conclusions were that the:

- F&CO made a mistake in believing that they has sufficient experience to manage the project themselves;
- F&CO had no formal project management methodology;
- project management methodology employed by the F&CO's consultants did not exercise control over project costs.

Case 5: Northern Ireland Education and Library Board

In 1984, a Computer Steering Group was set up to oversee development and implementation of an IT strategy. By March 1991, the cost of developing and implementing the strategy had risen from £6.7M to £15M. Software problems had delayed the payroll system by four years, while hardware that was purchased did not meet requirements and an addition £2M had to be spent on replacements and upgrades.

The conclusions were that the Board:

- failed to use a rigorous IT project management methodology;
- did not monitor and control project costs;
- inadequately specified their requirements.

Case 6: The Training and Enterprise Councils' (TECs) "Field System"

The TECs were created in 1988 by the Department of Employment, who were already developing the Field System for their own use. The Field System supported contract management (planning, monitoring; payments and accounting) and various office functions, and the Department decided that it should be used by the TECs to support their business. However, the Department experienced significant problems designing and implementing the new system, which in the view of many TECs failed to meet their business needs.

In April 1993, the Department decided to release the TECs from their contractual obligation to use the Field System and gave them responsibility for their own information technology. By then the Department had spent £48M on developing the Field System. An NAO survey later showed that most TECs had made little use of the system, or planned to replace it altogether within the next 12 months.

The NAO report highlighted insufficient project management skills for such an ambitious and technically complex IT project.

Case 7: Customs and Excise

In 1987, Customs and Excise introduced a new accounting system. By February 1988 over 4, 000 items valued at £900M lay uncleared in suspense accounts, and the Department were unable to present their accounts for audit.

The NAO report cited the:

- advice of consultants to bypass parallel running against the new system because it would be costly and slow down implementation;
- same consultants benefiting from their inappropriate recommendations by helping the Department to clear the errors that resulted, undertake a post implementation review and implement improvements.

Case 8: The Insolvency Service

In 1988, the Insolvency Service introduced a supplementary computer system to calculate fees, and in 1989, they began to use it to record transactions arising from the realisation of estates. The system did not work properly, and by March 1991, imbalances between the manually maintained cashbook and corresponding computer records totalled £13M. The problems cost £150,000 to rectify.

The NAO report cited:

- failure to exercise controls in transferring data from the old to the new system;
- lack of parallel running to test the integrity of the new system under operational conditions.

Case 9 : Wessex Area Health Authority

In May 1984, the Authority launched their regional information systems plan. This was intended to provide computer systems that would optimise the use of information to improve clinical and other health services. In April 1990, when the plan was abandoned, at least £43M had been spent by the Authority. District Health Authorities had also spent money on the scheme, but it is impossible to identify how much.

Following their examination of witnesses, the Public Accounts Committee concluded that:

- the Regional General Manager, who carried the main responsibility for the project, was a man of strong vision and a determination not to be deflected off course. He "presided over a series of actions incompatible with the proper handling of public money and without regard to clear evidence that the project was going badly wrong";
- there was a serious lack of communications between the Regional Health Authority management and higher tiers of management regarding the true status of the project as it progressed;
- Anderson Consulting were involved in both appraising available computer software and, later on, in supplying it (via a contract that contained no quality provisions and no maximum price). The PAC stated that it was clearly wrong for somebody who was tendering for a department's business to be advising them as their consultant. Similarly an Authority manager, who was on secondment from IBM, advised the Authority on the purchase of an IBM computer for a sum between £0.5 and £1M more than what it could have been purchased for;
- the contract for the supply of the IBM computer was placed without competitive tendering, contrary to the Authority's standing orders, as were

other valuable supply contracts, while there was also a failure to include performance targets and penalty in contracts;

- the Authority placed considerable reliance on consultants throughout the project, but did not control their activities;
- the Authority failed to keep control of their capital budget, while higher tiers of management did not know until 6 months after the end of the financial year that there had been a substantial overspend on the project.

2.3 Analysing the problems

The investigation of these IT project failures revealed a number of common problems, which can be summarised as follows:

1) **Failure to assess (and manage) project risks**, in particular stemming from **:-**
    - an unrealistic business case;
    - technology problems;
    - lack of user commitment to the project.
2) **Ineffective project management** :-
    - inexperienced IT project managers;
    - failure to apply, or an inadequate, IT project management methodology;
    - vague or incomplete specification of requirements.
3) **Miss-managing consultants and suppliers** :-
    - failure to seek competitive tenders;
    - vague terms of reference for consultants and open-ended contracts;
    - failure to monitor and control consultancy costs;
    - lack of independent quality assurance on consultants' work.
4) **System implementation failures** :-
    - unrealistic delivery deadline;
    - inadequate acceptance testing;
    - unworkable or non-existent contingency plans.

2.4 Lessons to be learned

None of the lessons revealed by the NAO's investigations are new. They are well understood in the industry, and most are a matter of common sense. They can be summarised as the need to:

a) agree and define an appropriate life-cycle methodology(s) to be used in the development of new systems;
b) use a sound project management methodology, and ensure that key project staff are trained and experienced in its use;
c) carry out risk assessments both before and during a project. In particular make a realistic assessment of :-
    - the organisation's ability to manage an IT project without expert assistance;
    - the time required to complete each stage of the project (estimates are often optimistic and assume that problems and errors will not occur!);
    - whether the assumptions contained in the Business Case are realistic, and how much tolerance on their accuracy can be accepted;
    - whether the Business Case continues to be viable as the project progresses;

d) monitor deliverables, including those from consultants, against targets throughout the project;
e) avoid significant changes to requirements (where they are justified they should, if possible, be included in future releases of the system);
f) continually re-assess the viability of the Business Case in the light of actual costs incurred and changing business requirements;
g) thoroughly evaluate assurances given by suppliers; don't take accept them at face value;
h) subject consultants to competitive tendering, and give them clear and precise terms of reference;
i) monitor the quality of consultants' work against their terms of reference (use independent external assurance if this is necessary) and agreed charges;
j) try to avoid staff turnover during the project, particularly among key staff;
k) do not economise by reducing training and testing (on average, about 20% of total development time should represent testing);
l) do not deliver training too early, and monitor the training programme to ensure that it is effective;
m) draw up workable plans to:
    - regress to the old system should the new system fail following cut-over;
    - provide effective business continuity over the long term.

These are examples the sorts of issues that the auditor will need to consider during the course of a typical IT project. Each will be covered during the course.

3. Feasibility Study and Project Initiation

3.1 Introduction

A project is a management environment set up to deliver a business product to a specified business case.

In other words, a project is temporary. It has a temporary organisation which only exists to deliver a product which is considered worthwhile to the business. Project products are wide-ranging and diverse in their scope. In a developing system project, they will include specifications, designs, manuals, software and hardware. In a large project, there may be many hundreds of such products to produce and control.

This chapter, and those that follow, provides a brief overview of the various stages of a developing system project, from its inception to its eventual closure. It also describes how a system development project is initiated and the steps that are necessary to establish whether a required product is worthwhile; and if so, the optimum solution for its delivery. Audit considerations are dealt with in the final section.

3.2 Identifying the need

During an IS Strategy Study, an organisation's needs for new or upgraded information systems are identified by undertaking a detailed examination of:

- legislative and/or business needs;
- any need to sustain or improve a level of service with limited or reducing staff numbers;
- the results of job analyses by internal staff or external consultants;
- the results of efficiency reviews;
- obsolescence of existing hardware, software or communications;
- the development of new or cheaper information system options.

A key output from the IS Strategy Study is the Catalogue of Information Systems Requirements. This lists individual requirements in their prioritised order of importance to the successful delivery of the Strategy. The need to develop or procure a new or replacement information system should stem from the Catalogue and represent a further step in implementing the IS Strategy.

3.3 The System Development Life-cycle (SDLC)

Projects are conceived and grow from a business need. But what seems to be a clear objective at the beginning can easily become blurred and confused as events progress. In the end the product which is delivered might not be what was expected and this may result in significant wasted investment.

Many risks can, to a greater or lesser extent, affect the successful development or acquisition of a new information system. These include the risk that the system will:

- never be delivered;
- be delivered late;
- exceed budget;
- divert user resources to an unacceptable degree;
- not deliver the required functionality;
- contain errors;
- be difficult to use;
- fail frequently during operation;
- not perform to the required standard;
- be difficult and costly to operate and maintain;
- ditto to enhance;
- not interconnect with other systems.

Where an organisation suffers one or more of these undesirable impacts, their cause can more often be attributed to poor management rather than to problems involving technology. Common management problems include:

- no standard approach to project management; in particular :-
    - unclear project definition, objectives and scope;
    - an inadequate Business Case to justify the project;
    - lack of communication :-
        - of clear responsibilities for line and project management.
        - between business, user and project management;
        - of project requirements;
        - of accurate and reliable information on costs and progress;
- failure to recognise over-ambitious objectives in relation to the organisation's capabilities to deliver them;
- lack of senior management commitment to the project;
- no full time and experienced project manager;
- failure to manage suppliers and consultants (who sometimes take control of the project);
- taking shortcuts due to lack of time (particularly cutting back on training, testing and quality reviews);
- failure to freeze the requirement specification resulting in slipping deadlines and rising costs;

- lack of quality standards;
- reliance on unproved technology.

The number of project failures can be dramatically reduced by breaking down a project into a number of more manageable stages each of which is to produce one or more pre-defined products. The advantage of a structured approach is that it helps to reduce the complexity of planning, monitoring and control. It also offers a number of points during the project where progress against pre-defined deliverables can be reviewed and corrective action taken as necessary (including abandoning the project!)

Collectively these stages, known as the Systems Development Life- cycle (SDLC), are:

- project initiation;
- feasibility study;
- requirements specification;
- system design (or external procurement);
- system development (or external procurement);
- user acceptance testing;
- system implementation;
- post-implementation review.

Each of these stages is described in this and the following chapters.

3.4 Responsibilities in the System Development Life-cycle

Historically, system development projects have often been undertaken by, or under the dominance of, an organisation's IT department. Where there is little or no end user involvement, the system that is eventually delivered often fails to meet its end users' expectations in full, and the organisation fails to reap the full return on their investment.

It is now widely accepted that the system's eventual owner and its end users must be identified before the project commences so that their needs are taken fully into account. Good quality end users should be assigned to the project team at an early stage and, where possible, the project should be under the overall control of a director or senior manager from the end user area.

Internal Audit and Security Management should also be closely involved in the project. This is particularly important during the system specification and design stages, as their input helps to ensure that IT security is adequately addressed. This need applies equally to bespoke developments and to the procurement of packaged software. The external auditor should be consulted on financial audit trail and data retention/extraction requirements, while legal and other expert advice may be necessary during contractual negotiations with suppliers. The external auditor's involvement is covered in more detail in chapter 8.

To summarise, the various roles and responsibilities during a system development project include:

Top Management: approves the necessary resources to complete the project, and provides commitment to it. This commitment from Top Management helps ensure involvement by those needed to complete the project.

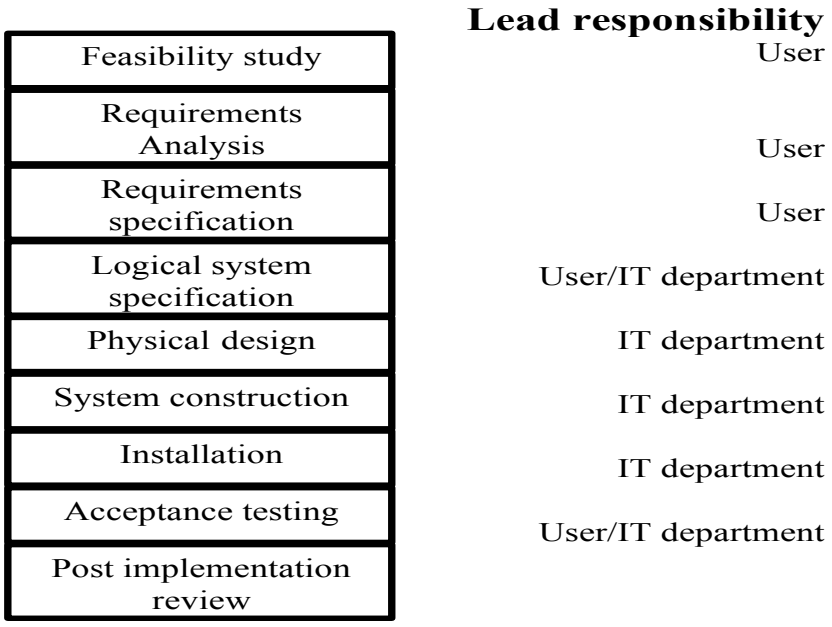
User Management: nominates the System Owner, although during development the system will be owned by the Project Board (on which the System Owner may well be chairman). The users should actively participate in defining requirements, the feasibility study, user acceptance testing and user training. The System Owner should review and approve system deliverables as they are produced.

IS Management: provides technical support for hardware and software environments by developing, installing and operating the required system. Provides assurance that the system is compatible with the organisation's IS environment and strategic direction. Assumes on-going operating support and maintenance activities after live implementation.

Project Board: group of people representing Business, User and Developer interests and who own the project. They are responsible for ascertaining on-going project viability and ultimate project success.

Project Manager: responsible to the Project Board (but not a member of the Board) for the day-to-day management of the project including monitoring and controlling project costs and timetable.

Figure 4 - life-cycle responsibilities



IS Project Team: completes assigned tasks, communicates effectively with users by actively involving them in the development process, works according to local standards, advises the Project Manager of any necessary project plan deviations.

User Project Team: completes assigned tasks, communicates effectively with IS by actively involving them in the development process, works according to local standards, advises Project Manager of expected and actual project plan deviations.

Security Officer: ensures that system design adheres to corporate security policies, consults on appropriate security measures that should be incorporated into the system,

tests system security prior to implementation, receives adequate training on any new security systems or procedures to assume security responsibilities after implementation.

Quality Assurance: reviews results and deliverables within each phase, and at the end of each phase, to confirm compliance with requirements and quality standards. The point where review occurs depends on the SDLC methodology used, structure and size of the system, and impact of potential deviations.

Internal Audit: reviews and reports to management throughout the project on the adequacy and application of the SDLC methodology. Assists in defining internal controls.

External Audit: for financial systems, advises on the measures that need to be included in the User Requirements Specification to ensure that the financial statements to be produced by the system will be capable of audit using accepted techniques.

3.5 The Feasibility Study

Where a major project is involved, a Feasibility Study might itself represent a significant project and require separate financial approval and project initiation. At the other end of the scale, work on the development of a low cost system which has no strategic impact and which is clearly feasible might commence without a formal feasibility study first taking place. It might also be the case that a project's feasibility is established during the IS Strategy Study.

The main objective of a Feasibility Study is to determine whether a proposal is viable and to recommend a suitable course of action if it is. The study might be undertaken by the organisation's own staff (from both the IT Department and the end use area(s)), by external consultants, or a mixture of both. It will:

- define the problem or need that requires solution;
- define broad or major requirements of the required solution;
- determine if a computerised solution is required or desired;
- determine if an existing system can be enhanced to correct the situation;
- determine if a commercial product offers a solution to the problem;
- for each alternative, provide an estimate of costs, benefits, technical and business risk, time-scales, and an assessment of the option's 'fit' or compliance with the organisation's IS Strategy.

During the Study, the Team should consult the Security Manager, the Internal Audit Department and the Legal Department on any requirements that may be necessary to satisfy the organisation's IT Security Policy or legislation, and which may serve to limit the number of options that are available.

The Feasibility Report might:

- endorse the proposal, identify a suitable solution to the problem, and seek authority to proceed with its development;
- recommend one of the following options:

- develop or acquire a demonstration system;
- ditto a pilot system;
- phased implementation;
- acquire a commercially available package;
- commission a turnkey system
- facilities management;
- outsource the service;

- recommend a hybrid solution requiring customisation of a commercial package (this is a high risk option and should be avoided if possible);
- conclude that the project would be feasible if its scope were to be reduced or were it to be split into a number of smaller projects;
- recommend that the proposal is impractical, too expensive, or unattractive for other reasons, and that it should not be taken forward.

Note: *phased implementation is applicable to projects that have long development and implementation time-scales, where changes in business and user requirements will naturally occur before they are fully implemented. Such projects should not be planned in a monolithic way, but be sub-divided into a number of self-contained phases.*

The Study will result in a decision by the IS Steering Committee on the course of action to be taken (including abandonment).

The contents and scope of a Feasibility Study will vary considerably between organisations due to their different circumstances and the constraints which apply to them. Annex 1 describes a model framework for a Feasibility Study Report.

3.6 The Business Case

If the recommendation of the Feasibility Study is to proceed with the project, the Study Team should also prepare a formal Business Case to provide full justification for this course of action. The Business Case should be submitted to the approving authority (e.g. the ISSC), using the Feasibility Study Report as supporting evidence.

The Business Case should be clear, convincing, well supported by evidence and fact based. The authorising authority should be convinced that the project would bring real benefits to the organisation before authorising it. In addition, they must also be in a position to assess all projects submitted for review by a common denominator. A standard framework should therefore be defined which should include the following:

- why the system is needed;
- the business objectives to be met or enhanced;
- any long term business implications;
- any staff and organisational implications;
- the alternative options which were considered;
- the recommended option;

- any important assumptions that have influenced recommendation;
- when the system can be delivered;
- its overall priority among other impending projects;
- the benefits to be delivered by the system;
- the risk of not delivering the benefits;
- the risks involved in pursuing the project;
- an investment appraisal (see next section);
- outline costs and plans for the project.

It will rarely be possible to estimate accurate project costs at this stage (in general, project costs tend to be under-estimated). The Business Case will therefore need to be kept under constant review as the project progresses in order to ensure that it remains viable. Where costs escalate and are not offset by a corresponding increase in the value of benefits, it may be necessary to reduce the scope of the project, consider alternative options, or abandon the project altogether.

3.7 Investment appraisal

In considering a range of options, it is important that managers understand which offers the best value for money in supporting business objectives. And in order to confirm that the chosen option delivers the expected benefits they must be identified in advance in order to provide firm basis for managing their delivery.

While it is relatively simple to identify costs accurately through accounting procedures, benefits (particularly those that are less tangible benefits) may be difficult to quantify in financial terms. Any benefits in improved efficiency and effectiveness should be identified, together with the Performance Indicators and Key Measures of Achievement against which their delivery can be measured.

For example when assessing the anticipated benefits of improving communications, a likely achievement target could be to halve the time taken from despatch of papers to receipt. In order to measure the achievement of this target, turnaround times before and after system implementation would need to be recorded and compared for a short period of time. Useful information may already be available in the form of measures obtained from previous similar IS/IT projects.

Once benefits have been quantified, it is important to present them accurately and clearly in order to make a convincing case. Two methods are commonly used:

Discounted Cash Flow: is the standard accounting method used within UK Government to show return on investment over a specified period. This is more than just a presentational tool in that it helps to assess the viability of a project by identifying net present value (**NPV**) figures which can be used to compare investment opportunities. The NPV identifies the speed and rate of return on the investment.

Narrative Text: any intangible benefits are usually reported on in the form of a narrative which examines how effectively the business aims would be met were the benefits to be realised. Using unambiguous language, such narrative can provide convincing evidence to support the Business Case. Narrative text should also be used to support the calculations and assumptions used to quantify benefits.

3.8 Identifying costs and benefits

The options identified during the Feasibility Study need to be listed, considered and discussed individually. It may be possible to eliminate some of them on grounds other than cost, perhaps because of their lack of strategic or technical 'fit', or for other practical reasons (e.g. a package vendor has a poor reputation for providing support). It is important to eliminate these options before undertaking a costs/benefits analysis of those options that appear viable.

In reviewing investment appraisal, auditors must be aware of the risk of 'double counting', which arises where more than one project claims a particular benefit in its business case. This has the effect of over-stating total benefits and distorting the overall business case. The converse occurs when one project attributes a cost to another, and vice versa, with the net result that neither project carries the cost.

There are many costs that need to be considered when undertaking investment appraisal. The following are the more obvious examples.

Development and implementation costs:

- hardware and software;
- communications facilities;

the staff resource necessary to deliver the system;

consultancy and contractors;

training on how to use the system.

Running costs over the expected life of the system:

- hardware and software maintenance support;
- communications maintenance and support;
- hardware and software replacement;
- ongoing maintenance and support of the live system;
- lease or rental of equipment.

In general benefits are far more difficult to identify and quantify. The following are some examples:

Benefits resulting from staff savings:

- savings in the number of staff employed;
- freeing staff to engage in otherwise unsupported activities;
- avoiding the need to increase staff numbers.

Benefits resulting from capital savings:

- avoiding the need for consultancy support;
- avoiding the need to provide training;
- savings on accommodation space;
- savings on maintenance costs;

- sale of redundant equipment.

Benefits which are less easy to quantify, but which may be significant and should not be ignored, include :-

- being able to do things quicker, resulting in improved marketability and competitiveness;
- improved staff morale, resulting in better productivity;
- systems which are easier to expand to meet future growth.

3.9 Project initiation

The aim of the Initiation Stage is to undertake the groundwork for the future management of a project and to obtain authority for it to proceed. Depending on the scale of the project, Initiation might relate to a study (e.g. a Feasibility Study), to the entire project, or to a single stage of a project (e.g. the Technical Design stage). An initiation stage might therefore occur at a number of points in the SDLC.

Some form of financial authority is normally needed before the Initiation Stage is reached. The level of authority necessary to approve the expenditure will generally depend on the value of the project, and on the organisation's rules on delegated financial approval. For major projects, this authority is likely to be provided by the ISSC.

During the Initiation Stage the following topics will need to be addressed:

- the scope of the project;
- appointment of the project's organisation and management (Project Board, Project Manager, Stage Manager and the Quality Assurance Team);
- appointment of staff and any consultancy support needed to undertake the project;
- any training needs which must be satisfied before the project commences;
- any options raised in earlier reports (e.g. the Feasibility Study Report);
- continuing validity of the existing user requirements, and of any assumptions and recommendations;
- the identification and management of both business and security risks.

3.10 Estimating

Good planning and control are crucial to the success of any IS project. Accurate estimates of timescales and resources are essential ingredients of any realistic plan. A major reason for project failure is the lack of realistic estimating.

All project stakeholders - both users and developers - should be involved in either producing or reviewing estimates. Although the preparation of estimates themselves may be carried out by a specialist estimator, the monitoring of a project's progress against the estimates produced is the responsibility of the Project Board, and the Project and Stage Managers. Project estimates, and the progress of projects against estimates, will also be of interest to managers and estimators working at tactical, strategy and business planning levels.

It is the responsibility of Project Boards to see that estimating is carried out. They must ensure that an estimator is assigned at project and stage planning points. The estimator must be given authority and backing to obtain the information required about the project needed to perform the estimating task. The estimator should:-

- become familiar with the proposed system;
- be aware of the environment in which the proposed system is to be produced/developed;
- prepare the estimate, using the appropriate estimating models and techniques(e.g. expert judgement - possibly based on the moderated view of a group of experts; analogy with similar projects; algorithmic models - often built into software tools; top-down/bottom up; trade indices);
- assess the effects that factors in the local development environment may have on the estimates;
- advise the Project Manager how to interpret the estimate, including the various risks;
- re-estimate as the project proceeds and when firmer estimates are possible.

3.11 The Project Initiation Document

The results of this work should be presented to the approving authority (e.g. the IS Steering Committee) in a formal report, the Project Initiation Document ("PID"), for them to consider and authorise. The PID should include details of:

- project objectives and time-scales;
- management, organisation and major control points for all stages of the project;
- all project deliverables
- technical and resource plans in sufficient details to calculate and allocate costs and resources for the project;
- quality criteria;
- risks to successful completion, and proposed ways of managing identified risks (Annex 1).

When formal authorisation has been obtained, the related task can commence.

3.12 Audit considerations

The purpose of the project, and its scale, will need to be taken into account when deciding exactly what needs to be reviewed at this stage of the SDLC. In general, the following questions should be considered:

a) in approving the project, were an adequate number of alternative options:
   - considered during the Feasibility Study?
   - presented to the approving authority?

b) was each option evaluated in terms of its business benefits, costs and strategic fit?

c) are the estimates of business benefits achievable and measurable, and have workable methods for measuring achievement been defined?

d) does the Business Case include the costs of staff training and of developing a Business Continuity Plan?

e) is the estimated pay-back period longer than the likely economic working life of the system? (e.g. 3-5 years micros; 5-7 years mid range; 7-10 years large mainframe)

f) does the viability of the Business Case rely heavily on long term estimates? (the risks associated with long term measurement periods need to be included in the project risk assessment)

g) does the cost/benefit analysis include appropriate tolerance (e.g. 10%) to take account of under-estimates of costs and over-estimates of benefits?

h) have project risks been identified, measured and considered by the approving authority?

i) has an appropriate authorising authority given formal approval for the project to proceed?

j) does the project clearly link with identified existing or future business needs?

k) does the formal approval to proceed specify the following?

- project objectives, scope and priorities?
- any major assumptions in planning the project?
- the end product(s) to be produced at each stage of the project?
- time-scales and deadlines project stages?
- budgets and resource allocations for each project stage?
- project organisation and responsibilities?
- arrangements for monitoring and reporting progress?
- ditto costs?
- the quality assurance criteria to be applied at each stage of the project?
- arrangements for assessing risks to the successful completion of the project as it progresses?

l) has a full time and experienced Project Manager been appointed to manage the project?

m) does the organisation have formal standards for project management, and are these being applied?

n) do project management standards specify the stages at which products are to be produced and progress reviews take place?

Annex 1 : Risk management checklist

Risk Management Checklist	Document No	
Other Headings	Status	
Project	Original Date	
Stage	Change Date	Version No
Project Manager	Department	
Stage Manager	Author	

The risk factors, i.e. those factors which affect the probability that the project will be completed on time and within the budget, and will deliver a quality product, come from four sources - project management, project staff, the nature of the project itself, and the maturity of the departmental management culture. These factors are itemised below, in the form of pairs of statements typifying low and high risk, on either side of a scale 1 to 4. One number in each of the scales is ringed to indicate my assessment of the risk attached to each factor. The ringed figure is multiplied by the weighting factor I have inserted in column (d) to give the figure in column (e).

(a) Low risk	(b) Scale	(c) High risk	(d) Weighting (suggested range)	(e) Total (bxd)
Project management 1 Full time, experienced project manager	1 2 3 4	Inexperienced or part time project manager	(5-7)	---*
2 Experienced and likely to be active participators	1 2 3 4	Management with little participation expected	(4-6)	---
Project staff 3 Users expected to be of good quality, actively involved, with relevant knowledge of the system	1 2 3 4	Little user involvement and little relevant knowledge expected	(3-5)	---*
4 High standard of supervision and narrow span of control	1 2 3 4	Span of supervision too wide and level of control inadequate	(4-6)	---
5 The technical team is experienced, of good quality and with appropriate skills	1 2 3 4	Inexperienced team lacking the appropriate skills	(2-4)	---
6 Staff are dedicated to the project	1 2 3 4	Staff have other responsibilities such as system maintenance	(3-5)	---
7 Low staff turnover	1 2 3 4	High staff turnover	(2-4)	---
The nature of the project 8 Staff are experienced in quality reviews and committed to their use	1 2 3 4	No quality reviews carried out in the past	(4-6)	---*
9 A typical system development cycle, with requirements definition, system specification, systems design, etc.	1 2 3 4	A system development cycle having no formal requirements	(2-4)	---
10 No unique or new features	1 2 3 4	Pioneering, new hardware or software, etc.	(2-4)	---
11 Current main operations will be affected minimally	1 2 3 4	Significant impact on mainstream operations	(3-5)	---
12 Hardware and software requirements determined and documents based on proven standards	1 2 3 4	Requirements not documented, or not based on proven standards; limited safety margins for contingencies	(2-4)	---
13 Little or no modification to existing application software	1 2 3 4	Extensive modification needed	(2-5)	---
14 Little or no other development work being undertaken concurrently	1 2 3 4	Other projects being developed concurrently	(2-5)	---
15 Little or no dependence on existing or developing systems not under the control of staff on this project	1 2 3 4	Dependent on other facilities not under the control of staff on this project	(3-6)	---
16 Project duration of one year or less, or small number of workdays compared with other completed projects	1 2 3 4	Project duration more than one year, or a large number of workdays	(2-4)	---
17 Little constraint on completion date beyond availability of resources	1 2 3 4	Mandatory completion date	(3-5)	---

(a) Low risk	(b) Scale	(c) High risk	(d) Weighting	(e) Total
18 Plans and estimates are based on reliable data	1 2 3 4	Planning and estimation data are unreliable	(3-6)	---
19 Investment appraisal and estimates prepared and well documented, using proven standards	1 2 3 4	Approximations used, or estimates not properly documented, or based on unproved standards	(3-5)	---
20 Suppliers are large, well established companies	1 2 3 4	Suppliers are new or one-man businesses	(2-4)	---
21 Few user departments	1 2 3 4	Several user departments	(4-6)	---*
22 The work affects few sites, which are easily accessible to the team	1 2 3 4	Many, or remote, sites are involved	(3-5)	---
23 Minor impact on user's current or future day to day work	1 2 3 4	Significant impact on users	(3-5)	---

The maturity of the departmental organisation

(a) **Low risk**	**(b)** **Scale**	**©** **High risk**	**(d)** **Weighting**	**(e)** **Total**
24 A well developed set of standards is in use	1 2 3 4	Few standards are available	(2-4)	---
25 A well defined quality policy exists	1 2 3 4	The quality policy is ill defined	(3-5)	---
26 Clear delegation of authority is practised	1 2 3 4	Centralised management with little delegation	(2-4)	---
27 Good relationship with Departmental Trade Union Side and with staff	1 2 3 4	Relations with DTUS and staff are poor	(2-4)	---
28 Brought forward from continuation sheet (attached)			---	---

Higher risk if greater than 286 (total of column (d) x 2.6)

Lower risk if less than 220 (total of column (d) x 2.0)

My assessment of the risk of this project (see notes 1f and 2 on page 4) is

Very high _____ Acceptable _____ } Tick

High _____ Low _____ } one

My recommendation for meeting the risks identified by a 3 or 4 marking against any of the above factors are attached (or in the Project Initiation Document if appropriate).

Signed .....Date .....

(Project Manager)

Notes:

1. The assignment sheets should be completed as follows:

   a) Ring one number in each of the scales 1-4 in column (b).

b) Assess a weighting for each of the risk factors, and enter it in column (d). A recommended range is shown in brackets for each category. Any weighting may be used, but the reason for any figure outside the recommended range should be recorded (see note 2 below).

c) Multiply each ringed number by the appropriate weighting, and enter the result in column (e).

d) Assess whether there are any additional risks not included in this assessment sheet. If there are enter them on a continuation sheet and assess them as in 1a to 1c above. Total the continuation sheet and add the total to page 3 of this checklist.

e) Total the weighting factors in column (d). Multiply the resulting figure by 2.0 to obtain the low risk limit, and by 2.6 to obtain the high-risk limit. Enter these limits against "low risk" and "high risk" respectively on page 3.

f) Total column (e) and enter the result on page 3.

g) Assess the risk of the whole project, bearing in mind the spread of marking in column (b), ant relevant departmental standards and experience with other projects. NB the risk factors marked with an asterisk in column (e) are regarded as critical to the success of the project. If any of them receives a marking of 4 in column (b), or if 2 or more receive a marking of 3 the project must be assessed as being of high risk, or very high risk, whatever the total score may be.

2. Any areas of high risk within the project, the reasons for selecting any weighting factors that fall outside the recommended range, and the overall assessment, should be recorded and itemised in the project initiation document, along with recommendations about action to counter the risk for approval by the Project Board. Thereafter, the degree of risk must be kept under review, to ensure that a low risk project does not become high risk without the change being noticed. The risk should be are assessed before each End Stage Assessment (except the last one) is held, and reported to the Project Board as part of the request to commence the next stage. All changes of assessed risk from the previous submission must be pointed out and commented on.

3. If assessment indicates that the project has little or no chance of successful completion, the IT executive Committee must be informed (by the Project Board).

Annex 2: Model format for the Feasibility Report

Introduction: provides details on the background to the study paying particular attention to the major reasons and the Study's terms of reference. Addresses:

- reasons for the study;
- terms of reference;
- study objectives;
- scope of the study;
- contraints;
- completion date;
- consultation
- management of the Study.

Management Summary: may be produced as a separate report. It gives brief details of the options considered, those rejected and that recommended, and plans for procurement and implementation. Addresses:

- recommendations;
- options considered but rejected;
- outline of design and development plans;
- procurement policy;
- implementation plans.

Study Approach: provides a brief overview of how the study was undertaken, and its cost. Addresses:

- an summary of how the Team undertook the Study;
- details of what the Study cost.

Existing Situation: describes the current situation in the area under review.

- business objectives, functions and processes;
- business area organisation, roles and responsibilities;
- current and potential future areas of strength and weakness;
- relationship with other business areas/organisations;
- existing information system support:
    - functions supported/not supported;
    - strengths/weaknesses;
    - technical opportunities/constraints.

Future IS Support: describes the role of the system in the IS Strategy, its scope and functionality. Addresses:

- scope and functionality;

- requirements targets (expressed as measurable):
    - essential basic requirements;
    - additional requirements;
- geographic distribution of information systems support;
- service performance required of the proposed system.

Proposed System: describes how the proposal described above can best be met. Addresses:

- the outline description of the system including: a system narrative, covering both manual and IT aspects;
- a system model, showing the major entities and processes;
- external interfaces.

Rejected Options: briefly describes rejected business and technical options. Discusses the advantages and disadvantages of each of the rejected options, as measured against the requirements target.

Financial Assessment: summarises all the costs of the proposed system, including the Feasibility Study. Addresses:

- summary of proposed system costs;
- comparison of proposed system with options;
- summary of expected benefits against current weaknesses;
- through-life discounted cash flow (DCF) for the recommended system;
- an assessment of how the recommended system would meet potential future needs, and also how it offers opportunities.

Project Plan: outline plans for the remainder of the project including time-scale for implementation, and the proposed management structure for project development and implementation.

Conclusions and Recommendations: based on previous sections of the Report.

Annex 3 : Example of two DCFs comparing the financial viability of two options

Option A

	Year O	Year 1	Year 2	Year 3	Year 4	Year 5
Costs Benefits	-300 0	-300 +100	-100 +100	0 +100	0 +200	0 +200
Total	-300	-200	0	+100	+200	+200
Discount Factor (6%)	1.000	0.9434	0.8900	0.8396	0.7921	0.7473
Present Value	-300	-188	0	+84	+158	+149
NPV	-300	-488	-488	-404	-246	-97

Option B

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5
Costs Benefits	-600 0	-100 +100	0 +400	0 +200	-300 +200	0 +200
Total	-600	+200	+400	+200	-100	+200
Discount Factor (6%)	1.000	0.9434	0.8900	0.8396	0.7921	0.7473
Present Value	-600	+189	+356	+168	-79	+149
NPV	-600	-411	-55	+113	+34	+183

It will be seen that Option B is preferred. Option A does not deliver any financial benefit even by year 5, while B becomes profitable by year 3 (NPV is positive), and by year 5 is showing significant financial gain.

4. Specifying user requirements

4.1 Introduction

This chapter provides an overview of the process for preparing the User Requirements Specification, and for obtaining proposals from suppliers where a system is to be provided externally. Audit considerations are covered in the final section.

4.2 The User Requirement Specification

If, following their examination of the Feasibility Report, the approving authority (e.g. the IS Steering Committee) decide that the project is to proceed, the next stage in the SDLC is to build on the user's outline requirements and produce an accurate and comprehensive statement of the user's needs, the User Requirements Specification (URS).

The URS is written in non-technical terms and consolidates all the material produced to date relating to the business functions of the required system. It represents the:

- formal declaration of the users' requirements of the proposed system;
- basis on which design work on the proposed system can proceed;
- basis for vendors to submit proposals and tenders;
- basis for acceptance testing the proposed system;
- basis of User Manuals when the system goes live.

The aim of the Specification Stage is to produce a comprehensive description of the current system, highlighting its deficiencies, describing how these are to be rectified by the enhanced or replacement system, and updated information on costs and investment appraisal (the Business Case must be kept under constant review as the project proceeds to ensure that costs are kept under control and the project remains financially viable). The completed URS should include the following information:

- a description of current problems, and how these will be resolved by the proposed system;
- a narrative describing what is required of the proposed system;
- main body of the document, containing:
    - outline description of business functions;
    - definition of data input;
    - definition of data outputs;
    - description of transactions;
    - description of processes;
    - description of current processing regulations;
    - description of major clerical functions;
- an outline description of interfaces with other systems;

- a description of management requirements covering security, change control, etc.

If a decision has been made to buy a system (or indeed a service such as facilities management), the URS should be sufficiently comprehensive to form the basis for advising potential suppliers in full of the organisation's needs, and enable them to respond with detailed proposals of how they propose to satisfy those needs. As a rule, the URS should therefore be written in such a way that it does not constrain the options open to either designers or prospective suppliers to provide innovative solutions by specifying exactly what technical solutions are to be employed in meeting the users' requirements.

It is important to ensure that the system's eventual Owner 'signs off' the User Requirements Specification to signify understanding and agreement before the project proceeds further.

4.3 System acceptance criteria

System acceptance criteria must be included in the URS. They will cover such features as a requirement for the system to meet a defined daily processing timetable, its maximum response times under full load (e.g. five seconds at any terminal), its ability to process specified volumes of data, security requirements, etc. Acceptance criteria are necessary to allow the developer or supplier to incorporate the necessary features and to provide a clear understanding of the basis on which the new system will be accepted.

System acceptance criteria will be referred to again in chapters 4 & 5.

4.4 Building the User Requirements Specification

Although the User Requirements Specification will be drawn up by the Project Team, it is often necessary to set up a number of multi-disciplinary teams (possibly including expert consultants) to consider and document all of the aspects of a large and complicated requirement. Building the Specification will involve extensive interviewing and analysis of the current system. It is essential that the system's potential end users are fully involved throughout this process as they will have a detailed knowledge of current procedures and of the business needs the new system is to satisfy.

Depending on circumstances other groups will also need to be involved in the process. These are likely to include the:

- **Personnel,** if the system is likely to need additional staff, or to result in staff redundancies;
- **Computer Operations** (or the computer bureau if the service is to be run externally) who will need details of the load the new system will place on existing computer and communications resources, and of work scheduling requirements;
- **Training**, in relation to staff and technical training needs;
- **Accommodation**, if the new system requires alterations to be made to existing room and building layouts, and/or places extra demands on the supply of utilities e.g. electricity, air conditioning, telephones);

- **Security**: who will be concerned with the protection of corporate information and computer equipment;
- **Legal Department**, on any legal constraints and requirements which need to be observed or included in the Requirements;
- **Internal Audit,** who should be involved in the identification of internal controls;
- **External Audit** (for financial systems), who will need to confirm that an adequate financial audit trail will exist, that the system will be well documented, and that it will be possible to extract data from the system on a selective basis to enable audit testing using CAATs.

During this stage, system analysts will probably construct a logical model of the system. This does not assume any particular type of software or hardware; instead, it comprises data flow diagrams, a logical data structure and logical processes which describe in conceptual terms the tasks the system will have to perform.

The advent of automated system development tools has resulted in a 'trial and error' development technique called Prototyping. This allows the user to see a working model of the proposed system on a computer terminal within a very short period of time. This can be assessed by the users and alternatives considered before the final choice is made.

Although Prototyping can result in faster development and a more polished product it tends to concentrate on the production of screens and reports, which are those aspects of the system most used by end users, and this can cause problems. By focusing on what the user wants and what the user sees there is a potential risk that the finished systems will have poor controls (e.g. security, audit trails, contingency). Also, extra functions not included in the original requirements specification tend to be added, and this can result in a functionally rich but inefficient system. And as such changes can take place rapidly, they are seldom documented or approved and can mushroom to a point of being unmaintainable.

4.5 System procurement: the Operational Requirement

Very often organisations - particularly those with little experience of system procurement - start looking for system solutions at the demonstration stage without first defining their requirements. The computer industry attracts some of the finest sales people in the world. They can be very plausible and will demonstrate the features of a product that they want to demonstrate. In order to gain an objective assessment of a system before those convincing sales people get involved, it is advisable to send an Operational Requirement to all potential suppliers and to evaluate their responses.

An Operational Requirement contains a complete statement of the procuring organisation's requirements. It is addressed to one or more possible suppliers of equipment or services, and is designed to draw from each supplier a "Proposal" describing in detail how the supplier could meet the requirements. It should start with a short general introduction which tells potential suppliers about the procuring organisation's business, how the requirement fits into its work, and its future plans. It should also describe the organisation's IS Strategy, how the Operational Requirement

fits into it, and if the proposed system is to replace an existing system, the existing facilities and any proposed changes.

The User Requirements Specification forms the main part of the Operational Requirement. In addition to the features described above, the OR should clearly indicate which requirements are "mandatory" and have to be satisfied if the potential supplier's proposal is to be considered. It should also specify the required format of the response, and any other general conditions which potential vendors must comply with. The skill with which requirements are expressed - and proposals evaluated - will be fundamental to the success of the project. Requirements need to be comprehensively and precisely identified, and expressed in a manner that allows them to be used to test the solutions offered by suppliers.

The criteria against which vendors' proposals are to be evaluated should be decided well in advance. For example, in evaluating a software package, it should be established that:-

- the software meets all mandatory requirements;
- the software is written to appropriate and acceptable standards;
- is adequately documented;
- the package ensures the completeness and accuracy of input, processing and output;
- the package permits a sound separation of roles to be maintained;
- the package provides adequate audit trails;
- backup and recovery of files and data meets user requirements;
- the package is compatible with existing hardware, operating systems and other systems (as set out in the technical policies which form a part of the IS Strategy);
- the performance of the package is acceptable in terms of ease of use, flexibility and efficiency of operations;
- the supplier provides comprehensive training;
- support and maintenance of the package are guaranteed and to a good standard;
- complete system documentation is held in escrow, and updated as necessary;
- the supplier is well established and financially stable;
- total ownership cost over the expected life of the system are acceptable (3-5 years micro; 5-7 years mini; 7-10 years) mainframe).

If a supplier's proposal is found to be generally acceptable, it should be considered for inclusion in the "short list"; in general this should contain three or four of the most attractive proposals. Short listed vendors are then invited to discuss their propose solution in greater detail, and to demonstrate that their proposal is both feasible workable. Again, it is essential to involve the prospective end users fully in this process.

Based on the outcome of the demonstration stage, an "Invitation to Tender" document should be drawn up. This is designed to draw a "tender" from the prospective supplier setting out precise commitments on costing, time-scales, functionality, etc.

4.5 Change Control

One of the biggest threats to the success of a development project results from inability to "freeze" the scope and requirements of the project, with the result that the Specification continues to evolve after the sign off has taken place. Changes to scope and requirements can result from revised business of legislative requirements (possible due to lack of, or an inadequate, IS Strategy); they may equally result from incomplete work during the Specification Stage.

Any changes to the User Requirements Specification represents a potential threat to the stability of a project. Minor changes are inevitable, although they should not be encouraged. However, major changes, particularly to the overall scope of the project, can cause real problems. Much delay can be involved while revised specifications and costings are produced and, perhaps, alternative system solutions re-visited. The business case will also need to be re-evaluated to examine the impact of the change on the viability of the project.

To minimise the impact and disruption caused by changes to the Specification, formal change control procedures should be adopted as soon as the Specification has been agreed and signed off. From then on, any request for change should be made the subject of a formal "change proposal". Change Proposals should be considered by the Project Board, and also any other interested groups who may be affected by it (this process is referred to as "impact analysis"). The Project Board might decide to:-

- implement the change as stated;
- implement the change, but in a modified form;
- implement the change, but in a future version of the system (known as "freezing" the change;
- reject the change and accept the problem or deficiency to which it relates.

4.6 The procurement cycle

In essence, the overall objective of procurement policy and procedures are to obtain the best value for money, which is usually achieved by competitive tender. The following are, in outline, the sequence of activities in system procurement that offer the best prospects of achieving value for money:

- specify what the user needs in an Operational Requirement and distribute it to prospective suppliers;
- consider the products that are available, and shortlist the most promising suppliers' proposals;
- examine the short-listed products and suppliers in detail; this will involve discussions, supplier demonstrations and visits to other users of the product;
- agree a valid technical solution with each supplier and produce draft contracts;
- ask suppliers to quote a firm price for the agreed solution;

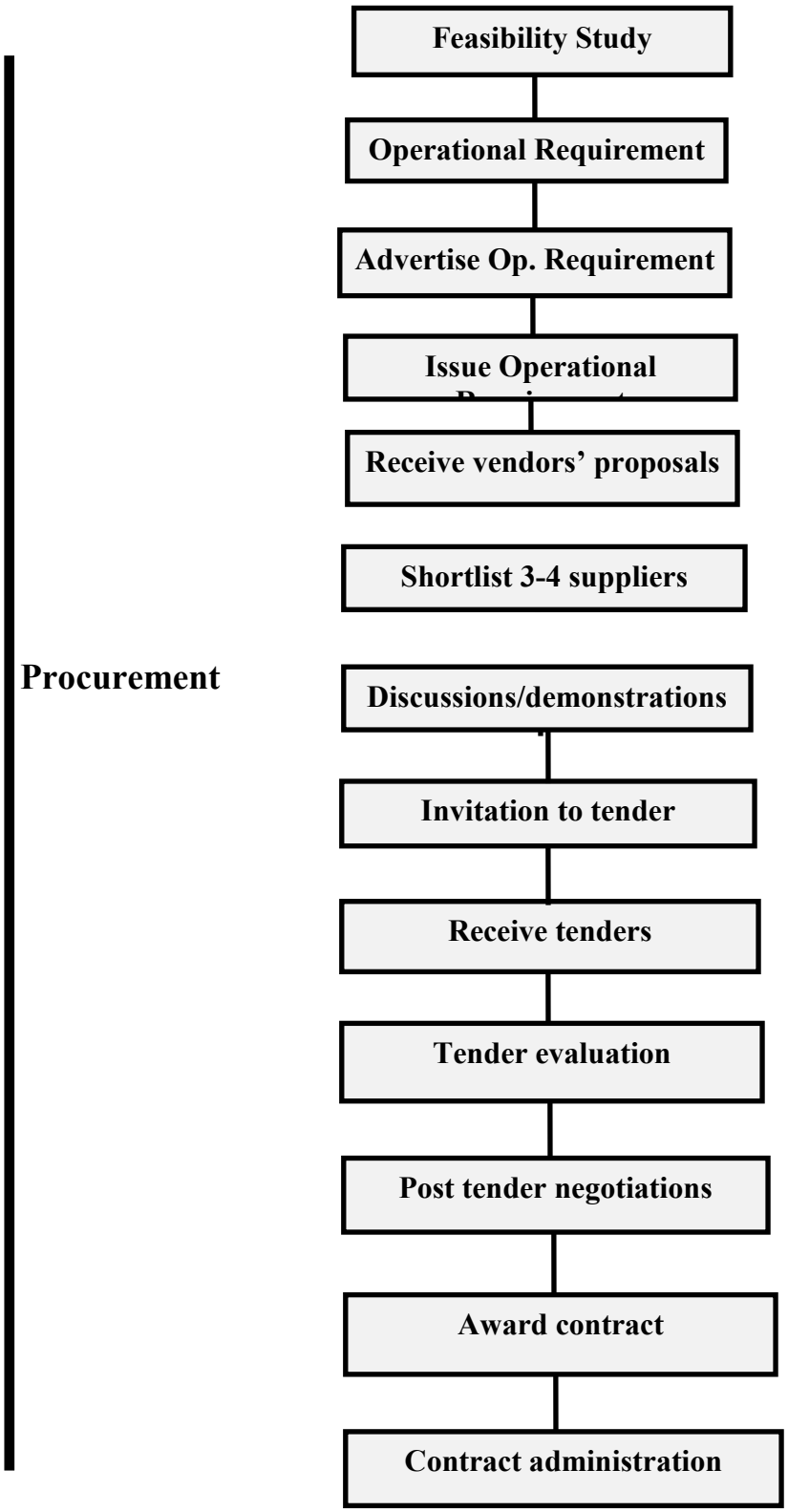
- evaluate the business deal that each supplier offers;
- choose the best combination of product/service, price and supplier (remember, the supplier's ability to both remain in business and provide reliable service over a number of years will be an important factor in reaching a decision).

In small or straightforward procurements, this process might be simplified.

In addition to needing a strategy for obtaining the best value for money from procurement, the SDLC method should set out procedures that ensure that the procurement strategy is properly implemented and cannot be subverted for personal gain. For example, a member of the procurement department might corruptly disclose to a prospective supplier the tender details submitted by competitors, or alter tender documents.

A checklist of the factors relevant to obtaining value for money from procurement decisions is at Annex 1.

Figure 5 - procurement overview



4.7 Systems purchased as packages

Packages are systems developed by computer manufacturers or software houses for the more common and widely used applications such as payroll, sales, purchases and general ledger. The facilities available within each package are fixed, but the purchaser can frequently select between the available facilities and vary the way in which they are used by means of parameters specified when the system is first set up. For example, a company buying a sales ledger package might be able to specify whether it wanted to apply prompt settlement discounts, if this facility was available within the package, and to set the discount rates applicable.

The company which purchases a package has no control over its development. Thus instead of detailed controls over the specification, design and testing of the package there should normally be procedures to select a package which best meets the requirements of the users and can be relied upon to work satisfactorily under operational workloads and conditions. These will typically include defining the users' requirements sufficiently clearly, usually in the form of a statement of requirements, and reviewing the actual performance of the package in use by other companies. In addition, there should normally be procedures for testing and implementing the system.

Implementation controls in respect of systems purchased as packages are therefore required in the following areas:

(a) Specification and selection

(b) Implementation and testing

4.8 The specification and selection of packages

Where a package is purchased from a software house or computer manufacture, it will be necessary to ensure that the package will meet the requirements of the users. This is normally achieved by preparing a statement of requirements describing the facilities required and the volume of data to be processed. The available packages are then assessed to determine which is most suitable, taking into account the facilities they provide and the volume of data they are capable of processing. The statement of requirements serves to enable the wishes of several users to be combined in a logical way and reviewed. It can then be used to communicate the requirements to potential suppliers an as a checklist in assessing alternative packages.

The preparation of a statement of requirements is similar to the preparation of a functional specification for an in-house system. The considerations regarding the involvement of all uses departments and the review and approval of the specifications are equivalent. The only significant difference is the usage to which it will be put. Instead of being used as a basis for preparing a technical specification and subsequently programs, the statement of requirements will be used to compare the different available packages and select the most appropriate.

Packages need to be compared carefully with the statement of requirements in order to identify any changes which must be made in order to meet the company's requirements. There is a strong argument for leaving a package unchanged wherever

possible, since this will reduce the risk of introducing errors into tested programs and will allow the company to take full advantage of the assistance provided by the suppliers, including any future improvements to the package that they may introduce.

However, in some cases a package may have to be modified where the best available package fails to meet important requirement. If this is done in-house, the control considerations already discussed in respect of the design and development of the modifications to a package. If modifications are made by the supplier then the company will need to specify its requirements clearly and carry out additional testing to ensure that the modifications have been properly made.

The selection of a package must take into consideration the possibility that amendments may have to be made to it in future to correct programme errors or to cater for changing requirements. For example, payroll systems, which are often bought as packages, have to be modified regularly to take account of changes in tax legislation.

The maintenance of a package is normally the responsibility of its suppliers unless the purchaser has made substantial modifications to it. It is necessary to assess the quality of the maintenance service provided. For example, it should normally provide for al future improvements to be made available, for amendments to be made at the users' request on reasonable terms and for a guarantee of continued support of this nature for the foreseeable future. Furthermore it should allow for the users of the package to make alternative arrangements for maintenance should the suppliers fail to meet their maintenance obligations for any reason, for example if they cease trading. It is common for the users to have a right to the source code and technical documentation supporting the package if this happens, so that maintenance can be carried out in-house or by another software company. Where the purchaser has made amendments, he will normally have to assume some responsibility for future maintenance.

Finally when selecting a package there should be procedures to ensure that it will work satisfactorily under expected operating conditions and workloads. The auditor is principally concerned that the package functions correctly for an accounting and control point of view. However management will wish for reasons of operational efficiency to consider factors such as the incidence of other program errors that may have no accounting implications, the speed and efficiency of processing, and the acceptability of screen or report layouts. The procedures in force to ensure that a package works satisfactorily will normally be a combination of reviewing the experiences of other users of the package and testing the selected package in actual operation at the company's offices, prior to its acceptance.

A review of the experiences of other users of a package is probably the best way in which the prospective purchaser can satisfy himself, prior to signing the contract, that the package will satisfactory. It involves visiting other users, observing the operation of the package and questioning management and staff about every aspect of the package. These include the facilities which it offers, its freedom from program errors, its speed and efficiency, how easy it is to use and the quality of the support provided by the supplier both during installation and thereafter.

Care should be taken to ensure that the most appropriate users are visited. The amount of useful information that the company can obtain from another user will depend on a number of factors including:

(a) the extent of the other user's experience with the package;

(b) the degree of similarity between the nature and size of the business of the prospective purchaser and the other user;

(c) Whether the other user uses the package in the same way as intended by the prospective purchaser.

The following questions are asked in the Audit regarding the specification and selection of packages and the availability of maintenance for them:

1. *Are there controls to ensure that:*

   (a) the systems and options selected meet the organisation's accounting and control requirements (e.g. statement of requirements prepared and approved by users and data processing personnel, trial periods);

   (b) any modifications required to adapt standard packages to these requirements have been identified, clearly specified and approved by users and data processing personnel?

2. *Does the contract with the vendor:*

   a. *provide for adequate support for the package over its expected useful life, including any necessary changes required by legislation and business developments;*

   b. allow the organisation to continue to use the package in the event of the vendor becoming insolvent, or indemnify it if the vendor proves not to have adequate title to sell/lease software originally produced by a third party?

4.9 The implementation and testing of packages

The packages are modified extensively in-house, the control considerations for implementation and testing will be the same as those relating to the in-house development of new systems. This section relates to the testing and implementation of a package which is not modified in-house, or where any in-house modifications are superficial, for example minor changes to screen or report layouts.

The testing of purchased packages can be considered in two parts:

(a) a review of the experiences of other users of the same package;

(b) independent testing of the installed package by the purchaser.

A review of the experiences of other users, as discussed earlier, will assist not only in determining whether the package will meet the purchaser's requirements but also in ensuring that the package has been properly tested and is reliable in practice. In assessing the degree of reliance which can be placed on the experience of other users of the package, it is necessary to take into account any changes which have been made to the package recently. These may have been either to correct errors or to add new features to the package. However, it is possible that they have introduced new errors which the other users have not yet detected. This may be because the other users have not received the latest version of the programs, or simply because sufficient time has not yet elapsed for all the errors to be discovered.

It is always important for the purchaser to carry out his own tests when a package is installed, even if the programs themselves have been extensively tested by other users. At the very least this is necessary to ensure that the installed equipment works properly and that the programs and files have been set up correctly. However when a major system is implemented it is usually desirable to test the programs themselves. The reason for this is that most packages are complex, allowing for different users to use them in different ways, and it is quite likely that no other users are using the package in exactly the same way as is proposed. Furthermore, most major packages undergo regular changes, which may introduce errors that take time to detect and correct.

The testing of a package by its purchaser corresponds to the user testing and live testing of a system developed in-house. Judgement must be exercised as to the extent of testing required in the light of the assurance gained from other users and the extent to which any program errors would be detected by user controls.

As with systems developed in-house, the implementation of a package requires the active participation of user departments, data processing departments and management, perhaps represented by a quality assurance department; for example, to ensure that appropriate clerical procedures are implemented. These control considerations have already been discussed in the section of testing in-house systems.

The following questions are asked in the Audit regarding the testing and implementation of purchased packages:

"1. Are new systems adequately tested (e.g. parallel or pilot running?

2. Are the following people involved to an adequate extent in the implementation of the system, including final approval where appropriate:

(a) users;

(b) data processing personnel;

(c) others (e.g. quality assurance)?"

4.10 Audit considerations

The existence of a formal User Requirements Specification and, if applicable, an Operational Requirement, are prerequisites for this phase of the audit. The auditor will need to confirm at an early stage in the project that these documents are to be produced and to a defined standard. If this is not to be the case, the auditor will need to consider the risks involved in the context of the project, and whether to raise the issue with senior management immediately.

In reviewing project documentation, the auditor will need to consider whether:-

a) all potential system users have contributed to the Specification, and that it reflects each group's particular needs;

b) data flow diagrams and conceptual designs also address user needs (some technical assistance may be needed in the evaluation of Systems Analysts' work);

c) all other interested parties have been consulted and that the Specification reflects their needs and advice;

d) adequate internal controls have been included in the Specification (see chapter 9);

e) an adequate financial audit trail has been specified and, in other respects (e.g. the provision of data extraction facilities), the system will be auditable;

f) the Specification and, where applicable, the Operational Requirement are comprehensive, well set out, and unambiguous;

g) the Specification remains in line with the project's scope, budget and deadline as approved in the original Business Case;

h) the Specification is in line with the organisation's technical policies and the IS Strategy;

i) the Specification has flexibility built into it to enable it to evolve in line with the organisation's strategic needs;

j) the eventual System Owner has signed off the URS to signify both understanding and approval;

k) where the system is to be procured externally, confirm that:-

* external acquisition has been approved by the authorising authority and an adequate procurement budget has been allocated;
* the URS and supporting details have been sent to a sufficient number of suppliers (e.g. where procurement of a commercial accounting package is concerned, the organisation should aim to evaluate at least six proposals);
* only essential features in the URS are described as "mandatory" (describing every requirement as "mandatory" is likely to either result in an over-elaborate and costly system, or severely limit the number of suppliers' responses to an operational requirement);
    - ♦ the chosen supplier's proposal conforms with the URS and supporting details;
    - ♦ adequate checks are to be made on the financial viability of the chosen supplier;
    - ♦ where software is to be procured, that full design details and supporting documents are held in escrow by an appropriate third party;
    - ♦ appropriate legal advice is to be obtained on the form of the contract document prior to signature.

Although procurement as a subject it is outside the scope of these notes, the auditor may wish to review the procedures for the procurement of computer systems to ensure that any legal requirements that apply to public procurement have been complied with, and that procedures have been designed to ensure that the organisation obtain good value for money. In particular that :-

a) the organisation's procurement procedures are fully documented and complied with;

b) there is an effective policy, or 'code of ethics', designed to prevent corrupt practices (e.g. trading terms and conditions such as cash discounts, free samples, services and other inducements are not diverted for personal gain);

c) procedures ensure fair competitive tendering;

d) tendering procedures ensure secure :-

* receipt of tenders
* custody of tenders
* opening of tenders
* recording of tenders
* tender evaluation (including the use of appropriately skilled personnel)
* acceptance of tenders

e) procurement contracts receive appropriate authority before they are let;

f) there is adequate segregation of duties between purchase and payments functions.

There are many contractual points which need to be considered both in the Invitation to Tender (ITT) and in the final contract. The auditor should review the contract to ensure that it <u>includes</u>:-

g) an adequately detailed Operational Requirement;

h) details of all hardware and software to be acquired, including reference to delivery, installation, testing, acceptance and disaster recovery;

i) all other services required, such as training, implementation support, manuals and other documentation, and maintenance;

j) a single contract with overall responsibility for provision and maintenance of software and hardware;

k) details of performance standards to be met, including a number of simultaneous users, anticipated transaction volumes (both immediate and future), terminal response times, up-time percentages, maximum period of down-time, and sanctions for failure to deliver;

l) details of maintenance levels and response times required; bug correction procedures including escalation steps; consider payment frequencies and times (quarterly in arrears preferred);

m) ensure that the performance requirements of the ITT will be warranted by the supplier;

n) acceptance testing procedures and time-scales;

o) payments terms (should be firmly tied to acceptance). Consider steps for the staging of payments and ensure retention to the end;

p) both detailed and summary implementation plans with target dates, a procedure for changing those plans (compliance with them should be compulsory);

q) ownership of property rights in the products, indemnities for breach of those rights and permitted use of any bespoke element (including a veto on sales or entitlement to royalties, if the purchaser will not have ownership);

r) key staff on both sides (and specify the Company's ability to reject the supplier's proposed project staff);

s) prohibition on the poaching of staff;

t) authorisation procedures for night/weekend work;

u) liability (negotiate reasonable limits or exclusions);

v) ownership of source code or provisions for mutually acceptable escrow agreement, for its deposit and testing;

w) circumstances that entitle either party to terminate;

x) dispute resolution procedures;

y) provision for regular progress meetings throughout the project, including making regular auditable reports on the progress of bespoke work.

There may be a number of standard clauses that the organisation's legal department would recommend be included in any contract, including matters such as confidentiality and force majeure.

Annex 1 : Checklist of factors relevant to obtaining value for money in procurement decisions

1. Status of suppliers

- financial viability
- design capability
- production capacity
- quality assurance status (e.g. ISO 9000) and track record
- delivery record

2. Supplier's proposal

- extent to which it meets minimum requirement
- design/artistic qualities, where these are appropriate
- compatibility with existing equipment
- 'extras' above minimum requirements which offer cost-effective advantage
- scope for improvements by later modifications or add-ons, etc
- conformity with standards (national, international, NATO, EU, etc)
- reliability - proven record?
- maintainability
- defect reporting and rectification arrangements
- repair/servicing arrangements

3. Immediate cost of acquisition

- initial price
- firmness of price (e.g. fixed; with or without variation of price; cost plus; etc)
- basis of agreeing price on associated or follow-on orders
- differences in cost-escalation formula
- foreign exchange risks and costs
- payment terms (e.g. on delivery; following acceptance; stage/progress payments)
- cost of financing interim payments
- financial guarantee requirements
- duties and taxes

- warranties and technical guarantees offered
- product liability arrangements
- scope for, and cost of, accelerating or delaying procurement

4. Delivery

- conformity with requirement
- reliability of offer
- operational and financial effects of earlier/later availability
- cost and trade-offs with stockholding costs at various locations
- liquidation of damages

5. Operating costs

- running costs
- cost of spares - present and future
- servicing and maintenance costs
- storage and other support costs

6. Product support

- quality of after-sales facilities
- ease of legal recourse to supplier

7. Replacement arrangements

- receipts from eventual disposal
- commitment to particular replacement equipment
- replacement time-frame

8. Strategic and structural

- safeguarding of vital source of supply
- length of the supply chain and its vulnerability to disruption
- offset considerations
- effect of procurement on price, availability and competition for future supplies (e.g. arising from dumping or artificially depressed quotations)
- effect on competitiveness of supplier
- encouragement of innovation offering improved value for money.

5. System design and development

5.1 Introduction

If the project involves building rather than buying a new system, detailed designs and specifications will need to be produced before work can commence on the development of an operational system. During these two stages of the SDLC plans will also need to be drawn up to cover other important activities, such as installing, loading and configuring the new system (often referred to as "system building"), acceptance testing, training and implementation. In a large-scale development project, these latter activities can be considerable tasks in their own right, and will need to be carefully planned and managed if they are to be successfully completed.

This chapter provides an overview of the main activities involved during the design and development stages of the SDLC, some of which are likely to be technically complex. The final section outlines the questions that the auditor will probably need to consider during these stages of the project.

Note: for the remainder of this section the term "configuration item" is used to refer to the various products produced during development, or procured externally, and which go to make up the operational system. Configuration items include the system's software, design documents, specification, manuals and equipment.

5.2 The Design Stage

The design of any system will inevitably involve some compromise between user requirements and various constraints over what can be achieved in practice. By taking a structured approach to the problem, particularly in separating logical and physical design activities, it is more likely that the optimum solution will be achieved.

During design, the Project Manager's primary concern will be to ensure that system designs match requirements as closely as is practicable and provide a sound basis for subsequent development work. System design activities are technically complex and although the users need to be involved at specific points, particularly in the design of user interfaces (screens and reports), most of the detailed design work will be undertaken by system analysts and other technical specialists.

Although designs will be based on the User Requirements Specification, this is a high-level specification and is unsuitable for system development purposes. System analysts will therefore need to translate it into detailed system designs and technical specifications. Prototyping techniques (covered in chapter on RAD) might be employed to demonstrate design options to the users, and to refine solutions.

Overall, the Project Team will need to address three broad objectives during the Design Stage. These are to produce:-

a logical design : this is a conceptual or abstract design that provides a solution to the user requirement;

specifications : define how each user requirement is to be met in terms of the hardware and software to be employed;

a physical design : maps the logical design onto the implementation environment (i.e. the hardware, operating systems, database management systems, programming languages).

The detailed technical designs and specifications produced during the Design Stage will then be passed to the programming teams for coding and development testing.

5.3 Designing the logical system

The overall aim of logical design is to produce a precise statement of what the physical system is to do, but independently of any particular operating environment. By concentrating on meeting user requirements designers can avoid the restrictions imposed by limitations in hardware and software. This approach also enhances "portability"; that is, the ability to transfer the resulting system from one operating environment to another.

The objects of the logical design stage are therefore to define the:-

- logical design of the process involved in terms of data flows;
- data structures necessary to support logical processing;
- contents of all computer inputs, outputs and system files;
- estimated file/database storage requirements;
- memory requirements;
- requirements for turnaround and response times;
- requirements for batch/interactive/network facilities;
- requirements for security and recovery of files and data;
- interface/compatibility requirements with other existing or planned systems;
- auditing requirements.

5.4 Designing the physical system

Physical design involves producing data and program specifications tuned so that they will comply with performance criteria. It involves translating the logical design into terms of the hardware, operating system, database management system, programming language, etc. which will be necessary to implement the system. In practice, these factors are likely to impose constraints on the way in which the logical design can be implemented, and will inevitably involve some degree of compromise.

The products to emerge from the physical design stage will include:-

- a narratives describing each process;
- logical flowcharts;
- file and database specifications;
- computer program specifications;
- computer stationery specifications;
- record specifications;
- screen and report layouts;

- menu structures;
- on-line help specifications.

The end user should be closely involved in the production of screen and report layouts, menu structures and on-line help facilities as these represent the interface between the system and its users.

5.5 The Technical Specification

Collectively, the User Requirements Specification and the Logical Design comprise the Business System Specification. The Technical Specification for the system initially comprises the Physical Design, but as the project proceeds, program specifications and designs, detailed forms designs and detailed procedure designs are added to it.

When completed, the Technical Specification will provide a complete technical definition of the system in terms of the specific computer environment (hardware, telecommunications, software, etc.) in which it will operate. The Technical Specification will also serve as an accurate technical description of the system throughout its life and provide the main source of reference for the staff responsible for developing and maintaining it. It should therefore be subject to configuration management (see below) and change control procedures in the same way as other important system assets, with an up-to-date backup copy being stored remotely as part of the organisation's Business Continuity Plan.

5.6 System development

During system development, the Technical Specification will be used to produce all the configurations items necessary to deliver an operational system. These include :-

- software;
- user and operational manuals;
- business continuity plans;
- and plans to cover activities such as system building, testing, training and implementation.

5.7 Configuration management

A computer system may easily contain many hundreds of configuration items. In general, these will comprise hardware and communications equipment, ancillary equipment (e.g. power supplies and air conditioners), application and system software, and a great amount of documentation. For example, behind the software lies specifications, designs, test data, user manuals, etc.

As requirements change and as errors are identified during testing, the configuration items that are affected will need to be identified and changed. Unfortunately the causes of change can arise in an unpredictable way and this makes control very difficult. Typical change control problems during development might be that:-

- a design change requires ten changes to be made to a group of software modules, but only nine are incorporated and one is forgotten;
- an error is reported, but it cannot be reproduced because it is masked by changes made since the item was released for testing;

- a revised version of a module is released for testing and the previous copy removed. The revised version contains errors but there is no way to re-introduce the previous working version to allow other aspects of testing to continue.

Overcoming these and many similar problems requires the discipline of recording configuration items and their relationships with each other; recording changes to them; and recording exactly which have been released for testing and operational use.

This discipline is called "configuration management" and should be under the overall control of a competent Configuration Manager during the development of the system, and throughout its working life. The Configuration Manager might be regarded as a sort of Librarian; in fact software tools that perform this function automatically are sometimes referred to as "auto librarians" (Annex 1).

In practice IT projects often produce hundreds of configuration items, often with complex relationships between them. When combined with a high rate of change (see below), configuration management cannot be performed effectively without the aid of software support tools. At the simplest level PC database and spreadsheet packages can be used quite effectively to assist with the task. But in a large scale project they would probably be inadequate, and the purchase of a commercial purpose built configuration management package may be justified.

5.8 Version control

A significant amount of testing and quality assurance will be carried out during the Development Stage. This will aim at establishing that individual software modules work as specified; that the linkages between related modules, programs, suites, and systems are correct; and that other configurations items are of acceptable quality.

As testing progresses each configuration item should be clearly segregated from those that have yet to undergo testing or quality assurance. And as mistakes during software specification, design and coding are inevitable, changes will need to be made, including changes to configuration items that have already been successfully tested or quality assured.

To avoid confusion, each configuration item should be allocated a unique "version" number to be updated whenever the configuration item is changed. This process is called "version control". It aims to ensure that different versions of the same configuration item are clearly distinguishable from each other, and is an important aspect of configuration management.

Compatible versions of different configuration items, which together comprise working system, are sometimes grouped together to form a "release". Configuration items which form a release are generally stored in one or more software libraries dedicated to that release alone. "Major" releases (e.g. Release 2.0) of a system generally contain significant new or enhanced functionality; "minor" releases (e.g. Release 2.1) generally remedy important defects which cannot be delayed until the next major release.

5.9 Change control

The importance of effective change control has already been mentioned in connection with managing changes to the User Requirements Specification. During both computer and non-computer testing, changes to configuration items will inevitably be

necessary as a result of late changes to the URS and to correct design and programming defects. Particularly in an IT project thT involves large-scale bespoke development, changes will arise in considerable numbers, and unless they are tightly controlled :-

- time and effort will be wasted on unnecessary changes;
- failure to set priorities will result in urgent changes being delayed by those that are non-urgent, and this is likely to delay the progress of system building and operational testing;
- unanticipated problems will occur in through failure to assess the overall affect that a change might have on other parts of the system;
- unauthorised changes might enter the system which could later result in breaches of confidentiality, fraud or other forms of system misuse.

The Project Initiation Document should state clearly the change control procedures that are to apply during the project. These will need to cover all configuration items - both produced by the project and purchased - and changes to the computer environments used for development and testing. For example, change control procedures will need to cover:-

- software under development, and related documentation (program designs, specifications, test plans, etc);
- other documents (training material, Acceptance Testing Plan; Implementation Plan, Business Continuity Plan, etc);
- standing files, master files, and databases used for testing;
- infrastructure software used during testing (e.g. database management system; operating system; communications management system; transaction processing system) together with system configuration files (e.g. tables of authorised users and their access rights);
- the computer hardware and communications equipment which makes up the test environment.

The principles that change control procedures should apply can be summarised as follows :-

- appropriate authorisation :-
  - all requests for change to be reviewed and authorised;
  - changes which require more than a specified level of resources, or which are likely to cause significant slippage in the project timetable, should be referred to the Project Board for approval;
  - on completion, all changes should be reviewed for compliance with change and configuration management procedures, and authorised for release;
- impact assessment; requests for change should be referred to other project staff for comment on their potential impact on the project overall;

- prioritisation : all approved requests for change should be costed (to assess their impact on the Business Case) and allocated a priority;
- trend analysis : involves classifying requests for change by type and urgency, and periodically reviewing change statistics to identify any emerging trends that might indicate a common problem;
- separation of roles :-
  - between change building and review/inspection/testing;
  - between change building/testing and change implementation;
- an appropriate level of quality review, development and operational testing;
- back-out plans; to cover failure of the change and permit the system to be returned to its previous state.

Obviously, how these principles are implemented in practice will vary considerably with the type and scale of the project. For example, in a small project it might be desirable that the Project Manager reviews and authorises all changes, but this will be neither possible nor desirable in a large project where many changes covering a wide diversity of topics are likely to occur. In this situation, a more elaborate structure (possibly comprising one or more Change Review Boards that meet periodically) will be necessary. But whatever change management structure is put in place, the principles outlined above should be applied.

5.10 Testing: overview

Testing commences during the design phase, during which designs and specifications should be subject to quality reviews (non-computer testing), and continues during the system development and acceptance testing phases of the SDLC.

Computer systems are tested to prove that they perform to the satisfaction of the various interested parties. This includes the developers, operations staff, and the end-users (including the System Owner). It may also include system administrators, security personnel, and auditors. In practice testing can only give reasonable assurance that all is well and that a system behaves as intended and as predicted; but it cannot give positive proof that any module/program/system is free from error. This is due in part to the extremely high number of possible program paths, and in part to the practical impossibility of generating test data that will adequately test all paths with all combinations of data.

The overall objective of the testing process is therefore to ensure that the delivered system is of adequate quality. To meet this objective it will be necessary to confirm that the new system:-

- conforms with the organisation's technical policies and standards;
- performs all the required functions;
- can be used by the staff for whom it is intended;
- meets it performance objectives;
- is reliable in operation.

The requirement to demonstrate that a system is reliable implies that it should be tested, not to demonstrate that it works, but to uncover as many defects as possible. Tests must therefore be designed that attempt to demonstrate that the system :-

- does not do what it is supposed to do;
- does what it is not supposed to do;
- is not operable by the staff for whom it is intended.

Other important principles that should govern testing - and indeed any quality control - activities are that there is :-

- no testing without measurable objectives;
- no testing without recording;
- no recording without analysis;
- no analysis without action.

Defects uncovered during testing might be corrected if they are considered to be of sufficient importance to justify the cost and time involved in taking remedial action. But it may be preferable to live with a defect if it is trivial, or defer remedial action until a more convenient time, for example by including a fix in a later release of the software. If a defect is corrected, the system (or perhaps parts of it) will probably need to be re-tested to ensure that the change has not introduced other unforeseen problems. This process is known as "regression testing".

5.11 Quality control

ISO 9000 defines quality control as the "operational techniques and activities that are used to fulfil requirements for quality".

As quality control is concerned with the quality of individual products produced during the project - in other words confirming that they fit for their intended purpose - it follows that it is the responsibility of the Project Manager to ensure that effective quality control is carried out. Quality control mechanisms include both formal and informal reviews, walkthroughs, testing, and inspection.

Quality control costs both time and money, and Project Managers are often tempted to dispense with it, particularly when working to an unrealistic, imposed deadline or where slippage has occurred in the project time-table. Removing what appears to be a "non-productive" activity apparently brings the project back on schedule. **This is a false economy**. It stores up greater problems for both the later stages of the project and for the maintenance and operations activities following project delivery. And there is growing evidence that quality control and productivity gains, far from being mutually exclusive, are complementary.

In practice, quality control can be an intricate subject, and the following paragraphs provide only a brief overview of the main types of quality control check. Students will need to refer to published text on the subject for more detailed information on quality reviews and software testing.

5.12 Quality reviews - the Fagan Inspection

Quality reviews cover various non-computer testing activities. For example, to determine whether a product is :-

- complete and free from cosmetic and mechanical defects;
- is correct (e.g. a specification or plan), is sufficiently comprehensive and is targeted at the appropriate skill level for each category of user;
- complies with relevant standards.

There are numerous approaches to reviewing quality in project documentation. That described here was developed by Michael E Fagan of IBM. It carries forward earlier ideas on statistical process and statistical quality control, and was originally used to inspect software code. It was later shown (primarily by Carolyn Jones of IBM) that any document could be inspected in this way.

The object of a Fagan Inspection is to ensure that all project documentation is clear, self-explanatory, unambiguous, consistent with other documentation in the same project and with the standards laid down in the organisation's SDLC method. It involves rigorous procedures that sometimes cause people, initially, to re-act against it. However, evidence of growing productivity and decreasing error rates demonstrates the effectiveness of this bureaucratic procedure where it is properly applied.

The fundamental base of a Fagan inspection is the database which contains defect-related data, for example :-

- type of defect (e.g. logic, standards violation, design error, etc);
- severity of defect (e.g. critical, major, minor);
- cause of defect (e.g. missing, wrong, extra);
- percentage of each of the above against page, time taken, organisation norm.

Such figures give precise management information, thus enabling effective action to be taken in problem areas. The following outlines the inspection process :-

Roles involved in an inspection:

Moderator : leads the whole process and must be trained in the technique. Chairs the defect logging meeting.

Inspector(s) : must be competent to assess the product from their particular viewpoint (i.e. user or technical). The inspector(s) take personal responsibility for the quality of the product.

Author : of the document being inspected and presenter.

Reader : a person who reads aloud the whole document at the defect logging meeting and has no other responsibility in the process.

Action team : take forward the results of the casual analysis meeting (see below).

Process :

Entry condition check : does the document appear to meet the laid down quality criteria (i.e. is it ready for inspection?).

Kick-off meeting : attended by all involved in the process to ensure a common understanding of what the document is trying to achieve and sets the rules for the defect logging meeting.

Preparation : inspectors study the document independently and note defects.

Defect logging meeting : the object is to agree a list of defects in the product being inspected. Statistics are noted. Minutes are not taken, only defects are logged.

Casual analysis meeting : the statistics are reviewed and compared with organisation norms.

Rework : author corrects all defects.

Exit condition check : has all rework been done satisfactorily?

Exit : product is base-lined (i.e. brought under formal configuration management and change control).

5.13 Development testing

Software testing is a further aspect of quality control. It falls into two broad categories; “development testing” and “operational testing” (see Annex 2).

During the development stage, the Technical Specification is used to produce or acquire the actual configuration items necessary to deliver an operational system. As these become available, they will need to be tested by the development team to confirm that they are of acceptable quality. This process is called “development testing”, and generally comprises two broad areas of testing:-

Unit testing: is the testing of an individual program module in an isolated environment before combining it with other modules to form a program. The objective is to determine whether the module is capable of accepting specific input and producing the correct outputs. Unit testing is normally carried out by the programming team leader. Program testing follows a similar objectives, but with all the modules in place to form a complete program.

Integration testing : is the process of adding new programs to an evolving system and testing to find :-

- errors in the interfaces between programs (sometimes referred to as "link testing");
- discrepancies between the program functions performed and those specified;
- that no unspecified functions are performed.

Because of its highly technical nature, the users are not generally involved in development testing. However, the users should be involved in "operational testing" which is described in the next chapter.

5.14 The Acceptance Testing Plan

Testing needs to be carefully planned and scheduled if it is to be fully effective. The Acceptance Testing Plan is therefore an important product from the development stage.

Acceptance testing (described in chapter 5) should not take place until development is complete. It provides a complete end-to-end test of both the computer system and its interfacing clerical procedures. From the auditor's perspective, acceptance testing provides a final check on whether the system is auditable (in terms of the adequacy of its audit trails, documentation and data extraction facilities) and is well controlled.

Initial thought needs to be given to what acceptance criteria are to apply to each requirement as the User Requirements Specification is being assembled. These criteria are gradually refined into a plan during system design and development, when it becomes clear exactly what sort of tests will be needed to prove that the delivered system is of acceptable quality.

Acceptance testing is therefore based on an analysis of the User Requirement Specification and any other acceptance criteria defined during design and development. This should aim to identify which requirements, facilities and functions are to be tested (and any which are not), their relative importance, and the method of testing to be adopted for each. This information is incorporated in the Acceptance Testing Plan which should document :-

- individual roles and responsibilities during testing;
- configuration management procedures (there is ample scope for confusion between different versions of software under test, and test data);
- expected versus actual test results;
- requirements for completing test failure reports;
- requirements for prioritising test failures (see Annex 3);
- change control procedures to be applied to the resolution of defects;
- procedures for re-testing following changes;
- requirements for signing off test results.

Individual tests should clearly show :-

- test objectives and evaluation criteria;
- staffing requirements to perform the test;

- the required test environment, including hardware requirements (e.g. the number of terminals or printers needed) and any particular system data conditions or standing data requirements.

The procedures and data for each test (referred to as "test scripts") also need to be documented. Test scripts comprise a set of systematic instructions to the person performing a test together with the test input and output data. Each test script describes what is to be done, who is to do it, when it is to be done, what to look for, and what to record. The input data specified in each script is that prepared specifically for testing the system, while the output data is a written forecast of the exact data that should result from the test.

It is also important to define the procedures for recording, reporting and fixing faults, and for managing changes to the User Requirements Specification and to any other important system documents (e.g. data flow diagrams, flow charts, etc.)

When the Acceptance Test Plan has been drawn up, the prospective System Owner should sign it off to signify understanding and agreement.

5.15 Other development activities

Other activities typically undertaken during this stage of the SDLC include the development of :-

system documentation: staff will need clear and concise instructions on how they are to use, operate and maintain the system, from both an end user and a technical support perspective. Manuals should comply with a formal documentation standard (a technical policy on documentation should have been defined during the IS Strategy Study) to ensure that they cover all necessary topics. The British Computer Society recommends that the following documents are produced:-

- user's manuals containing instructions on how to run the system;
- details of internal control procedures to be complied with in financial and accounting systems;
- additional system instructions on the technical aspects of computer and network operations;
- a system flowchart and/or narrative describing the purpose, functions, job scheduling of the system;
- a list of all jobs to be processed;
- a list and description of all databases, files and libraries;
- a list of all input requirements, inclusive of data classifications (for security);
- a list of the reports to be produced by the system, when they are needed, by whom and for whom;
- details of backup requirements;
- details on the administration of any access controls that must be maintained by the user and the data processing functions;

- details of all responsibilities and accountabilities for the operation of the system and its effectiveness;
- details of user clerical procedures including control procedures;
- details of interfaces with other systems including control and security procedures.

training plans: documentation will need to be produced to cover the delivery of training to a variety of potential audiences including:-

- end users in various functional areas of the organisation;
- middle and senior managers, who may need to use different features of the system (such as management reports and database interrogation facilities);
- technical support staff who will need to operate and maintain the system.

service level agreement: is an agreement between the user and the service provider who will be responsible for maintaining the software and providing the IT service. Work should be started on the operational characteristics that will be expected from the new system, such as:

- the hours during which it will be available to users;
- ditto the days;
- the required availability at any terminal and location;
- maximum acceptable downtime on each type of service failure;
- procedures to be followed in the event of system failure;
- Help Desk support;
- contingency requirements;
- how the agreement is to be monitored and reviewed.

implementation plans: should address the creation of the environment in which the system is to run, ensure that it is adequately tested and is functionally acceptable, and define the way in which the new system is to be introduced into production use. The plans must cover:-

- installation of hardware;
- any environmental or building requirements;
- the transfer of data from existing to new systems;
- setting up the Help Desk;
- backup and standby arrangements;
- cut-over from old to new systems.

5.16 Back-up and standby during system development

System development and configuration management libraries should be backed up regularly, and copies of the back-up stored in a secure remote location. Where failure to meet the system implementation deadline could have a severe impact on the organisation, it will also be necessary to make plans to ensure that development can

continue with the minimum of delay should a major computer failure or disaster occur.

5.17 Audit considerations

Some of the activities that take place during system design and development are technically complex. If the auditor intends to carry out detailed reviews of, for example, logical design, it will probably be necessary either to employ expert assistance, or to undertake training in the particular technical skills required.

The following are the general questions that the auditor will need to consider during this stage of the SDLC :-

a) does system design follow a defined and acceptable standard?

b) are completed designs discussed and agreed with the users? (perhaps with the assistance of prototypes - see Chapter 8);

c) does the project's quality assurance procedures ensure that project documentation (e.g. design documents, specifications, test and installation plans) is reviewed against the organisation's technical standards and policies, and the User Requirements Specification;

d) do quality reviews follow a defined and acceptable standard?

e) are quality reviews are carried out under the direction of a technically competent person who is managerially independent from the design team;

f) are auditors/security staff invited to comment on the internal control aspects of system designs and development specifications?

g) are statistics of defects uncovered during quality reviews and other forms of quality control maintained and analysed for trends? Is the outcome of trend analysis fed back into the project to improve the quality of other deliverables?

h) are defects uncovered during quality reviews always corrected?

i) does the production of development specifications also include the production of relevant acceptance criteria?

j) has a Configuration Manager been appointed? Has the configuration management role been adequately defined?

k) are all configuration items (hardware, software, documentation) that have passed quality review been placed under configuration management and version control?

l) has sufficient IT (in the form of spreadsheets, databases, and specialist configuration management support tools) been provided to assist with the configuration management task?

m) are effective procedures in place for recording, analysing and reporting failures uncovered during testing?

n) are effective change management procedures are in place to control changes to configuration items?

o) has a System Installation Plan been developed and quality reviewed?

p) has a Training Plan been developed and quality reviewed? Has sufficient time and resources been allocated to its delivery? (to avoid "skills stagnation", the delivery of training will need to be carefully scheduled);

q) has an Acceptance Testing Plan been drawn up? Is it to an acceptable standard? Does it cover all aspects of the User Requirements Specification?

r) does the Acceptance Test Plan clearly allocate roles and responsibilities for undertaking and reviewing the results of acceptance testing?

s) has the Acceptance Test Plan been discussed with, and signed off by, the prospective System Owner?

t) is the system development environment is regularly backed up with copies of backed up configuration items held securely at a remote location?

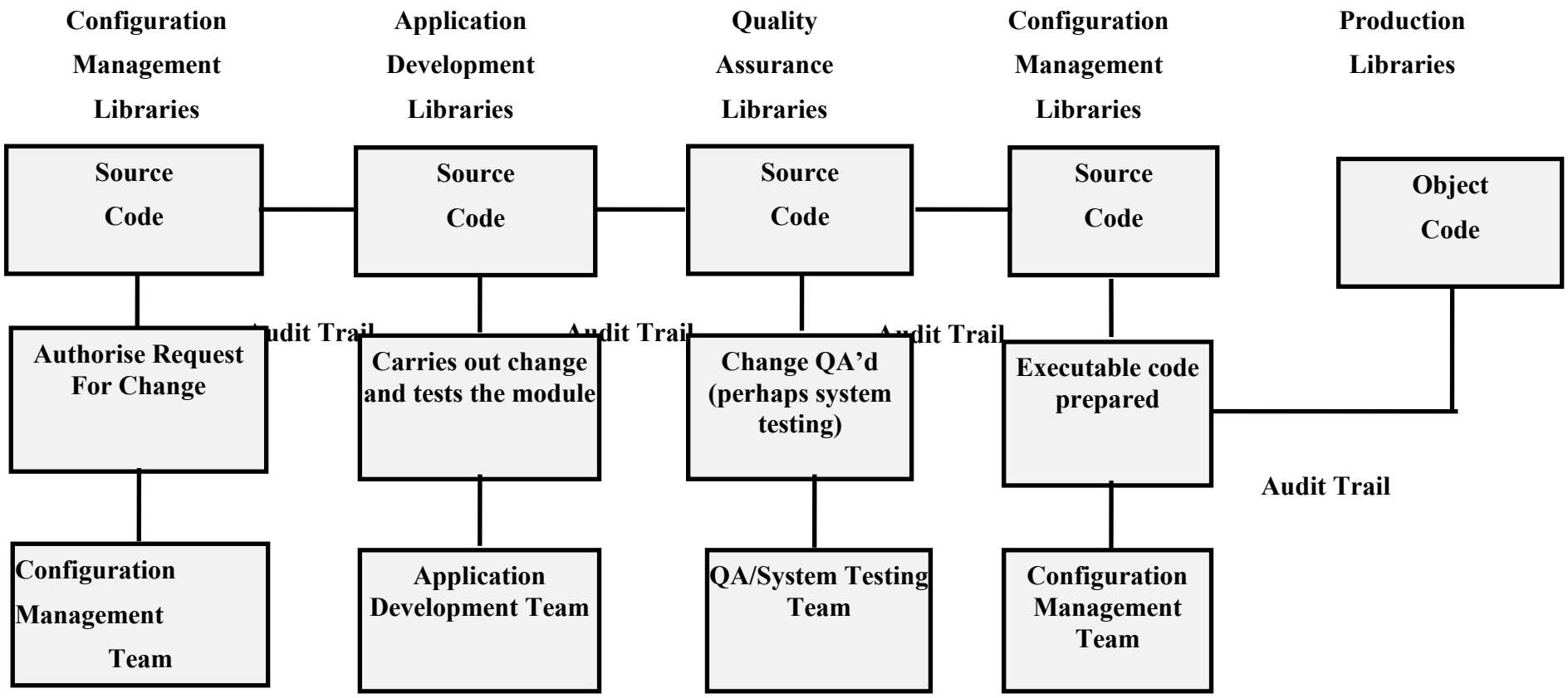
u) has the development environment been recovered from backup media?

v) are contingency plans commensurate (in terms of time to implement) with the criticality of the project?

w) do regular Project Board meetings take place to review project progress against budget and deadline?

x) is the Business Case regularly updated to ensure that the project remains viable?

Annex 1 : configuration management (controlling changes during development)



Annex 1 (con't) : Configuration management : controlling changes during development

1. In controlling change, the aim of configuration management is to ensure that the correct configuration item is changed; that the change is properly authorised; that it is adequately tested; and that the updated version is brought into production use at the correct time. Also, that relevant documentation is also updated to reflect the change and to provide an audit trail.
2. Exact procedures for achieving these ends will vary between organisations, with the type of development method involved, and also with the scale of the change (an altered page heading is unlikely to attract the same level of testing as the implementation of amended taxation regulations). The following are typical procedures.
    a. In response to an authorised Request for Change, the Configuration Manager copies the current source code, together with associated documentation, to the applications development library, and records the transfer. The Configuration Manager also takes steps to prevent these items from being released to any other development team while the change is outstanding.
    b. The applications development team make the necessary changes to the test version of the module, and test the changes.
    c. On completion of unit testing, the Quality Assurance/System Testing Team are notified that the module is ready for review. They copy it to their own library to prevent any further access by the applications development team.
    d. The QA team identify differences between the amended module and that in the configuration management library using a file comparison utility. The comparison report forms the basis of QA, and provides an audit trail of the changes that have taken place against the authorised Request For Change. Depending on the nature of the change, system testing may be necessary.
    e. On successful completion of QA the Configuration Manager transfers the source to the configuration management library, prepares the executable code and transfers it to the production system.

Annex 2: stages of software testing

	UNIT	LINK	SYSTEM	INSTALLATION	ACCEPTANCE
Main purpose	To find faults in modules or programs	To find faults in interfaces between modules or programs	To find faults in fully integrated systems	To find faults in stalled configurations	To gain formal acceptance by users
Tested against	Program specifications	Design specifications	System specifications	Installation specifications	Acceptance test specifications
Performed by	Programs	Programming Team	IT Services	Installation team Operations staff	Users
Reviewed by	Programming team	S/W development management	IT Services, users and developers	Installation team Operations staff	Users and IT Services
Designed at	Detailed design/program specification stage	Architectural design and specification stage	Requirements analysis	Requirements analysis and operational considerations	Requirements analysis
Test running can start	When unit is coded	When units to be linked have passed unit test	When link test complete	After system testing complete	After system testing complete

Development Testing **Operational Testing**

Annex 3 - example of an acceptance testing "incident priority table"

Priority	Description
Critical	Fix immediately - it is impossible to continue testing until this incident is resolved. The trial will be abandoned and re-run on a new date.
Major	Fix before test complete. This is a serious defect or non-conformity, but testing can continue. The test will be re-run when the defect is corrected, but in the event of a large number of major faults the Trial Representatives may decide to abandon the test and re-run on a new date.
Minor	Fix before system acceptance, or by a specified date or event.
Hold	Hold for later resolution. Hold (together with 'track' and 'new') is an Issue, Observation or Comment requiring further attention, clarification or discussion but does not interfere with the test process directly. This may be unclear documentation, confusion with the operation of tests or software, or similar incidents.
Track	Track for recurrence - this appears to be a non-repeatable incident.
New	New incident that appears to be serious, but insufficient data available for analysis. Further investigation required.

6. System building and acceptance testing

6.1 Introduction

The objectives of this stage of the SDLC are to build the environment in which the new system will run, and to ensure that the new system is of acceptable quality. Users should be closely involved, particularly in transferring data from existing systems to the new system and during acceptance testing.

Note: for the remainder of this section the term "configuration item" is used to refer to the various products produced during development, or procured externally, and which go to make up the operational system. Configuration items include the system's software, design documents, specifications, manuals and equipment.

6.2 Building the system

Building the system involves assembling configuration items to form a working system. The full extent of system building activities will depend on the size of the computers involved (PCs, mini-computers, mainframes), whether the system is to be run on new or existing equipment, and whether the system is centralised or distributed.

Typically, building a new system will involve the following tasks :-

- installing and acceptance testing computers and communications equipment;
- creating disc partitions and library structures;
- loading and configuring the system and infrastructure software (operating system; transaction processing system; database management system; security system; communications software; job schedulers; job accounting software, etc.);
- loading and configuring the applications software;
- creating files and databases;
- loading/transferring standing and masterfile date;
- introducing business and technical users, and setting file access permissions.

6.3 Data conversion

Some data will probably need to be transferred to the new system from an existing system(s) to avoid the high cost of re-creating it manually. This task must aim to ensure that data is transferred completely and accurately, and the audit trail preserved where necessary.

The practical problems involved in data conversion should not be under-estimated, and project plans should allocate ample time and resources to this task. The following are typical of the sorts of problems that might need to be considered when drawing up the Data Conversion Plan :-

- data might be subject to rapid change during the conversion process and before cut-over to the new system;

- in an accounting system, the new chart of accounts will probably differ significantly from the old and involve the need to produce complex mapping between the two (and designing the new chart of accounts is one of the largest operations in installing a new financial system because of the expert business knowledge that is required);
- a number of geographically dispersed servers might replace a central database with the result that data is distributed over several locations;
- the quality of the data in the existing system might not be of a high standard and need correction before it can be transferred to the new system;
- it is possible that the new system requires data items that do not exist in the existing system(s), and these will need to be created.

Transferring data in any quantity will almost certainly involve the need for purpose-built software. The supplier of the database management system may supply generalised loading packages, but it may be necessary to develop (possibly complex) bespoke software routines for the purpose. The production and testing of data conversion software will first need to be considered during the Specification Stage of the project.

Loading a new database system often reveals inconsistencies and inaccuracies that were not apparent in the existing system. As a result, much time might be needed to "clean up" the existing data and convert it to a form in which the new system will accept it.

To help ensure that data is transferred completely and accurately, and posted to the correct account, it is advisable to draw up a Data Conversion Plan and place the overall task under the supervision of a manager from the user domain; indeed, where there is a significant volume of data to be transferred to the new system, the task might be managed as a separate project with its own Project Manager and team.

Where it is possible to compare the outputs from both old and new systems, a period of parallel operation (see below) can provide strong assurance that data conversion has been correctly carried out.

6.4 Operational testing

As mentioned in the previous chapter, testing falls into two broad categories; development testing and operational testing.

The purpose of operational testing is to demonstrate that the new system will operate successfully within the live environment. It should not commence until both development testing and system building have been signed off. Overall, operational testing is designed to answer two questions :-

- "did we build the system right?" (i.e. according to specification)
- "did we meet the users' requirements?"

In general operational testing comprises three types of testing :-

system testing : aims to demonstrate that a new system meets the agreed requirements, and does so under all operating conditions;

installation testing : aims to demonstrate that the system is correctly installed in its live environment and meets the requirements of computer operations staff;

acceptance testing : aims to gain acceptance from the users that the installed system meets their requirements not only in terms of functions, but also in terms of performance and usability.

The remainder of this chapter deals with acceptance testing. This will be of particular interest to the auditor as it demonstrates, among other things, the extent to which auditing requirements (i.e. audit trails; data extraction facilities; internal controls - covered in chapter 9) have been delivered in the new system.

6.5 Acceptance testing

Acceptance testing is a complete end-to-end test of the operational system including all manual procedures. It aims to provide the system users with confirmation that:-

- the User Requirement Specification (including system performance criteria) has been met;
- end user and operational documentation is accurate, comprehensive, and usable;
- supporting clerical procedures work effectively;
- Help Desk and other ancillary support functions operate correctly and as expected;
- back-up and recovery procedures work effectively.

The bulk of acceptance testing should be undertaken by the end users supported by IT staff and expert consultants as necessary, and should continue until no errors or shortcomings remain. In practice, there is often a pressing need to meet an implementation deadline, and it is likely that the resolution of non-urgent problems may have to be postponed until a later release of the system.

Appointing a suitably experienced manager and following a pre-defined Acceptance Testing Plan will help to ensure that testing is effective.

In addition to testing system functions, acceptance testing must also test responsiveness (performance criteria should have been defined during the Specification Stage) particularly across the telecommunications links. These can be difficult areas to test because the live system often needs to be used in order to replicate true operational conditions, and this may have an impact on the organisation's live workload. Performance testing should address:-

average response time: usually defined as the time between the user depressing the transmit key, and the first character of the reply appearing on the screen, with a further maximum time specified for the screen to be completed;

maximum response time: the response time that must not be exceeded;

other response times: for example the time to:-

- load an application;
- load a major application;
- accept or move between fields on the screen;
- perform a single update;
- perform a multiple update;
- run a complex enquiry

volume testing: subjects the system to heavy volumes of data to test whether it can handle the volume of data specified in a acceptable time-frame;

stress testing: subjects the system to heavy loads or stresses (a heavy stress is a peak volume of data encountered over a short period).

Depending on circumstances, other types of acceptance testing which may be undertaken at this stage include:-

security testing: attempts to subvert the system's security and internal control checks;

clerical procedures checking: aims to confirm that all supporting clerical procedures have been documented and work effectively;

back-up and recovery: aims to confirm that software, configuration files, data and transaction logs can be backed up, either completely or selectively; and also restored from backup;

parallel operation: covered in chapter 6.

On satisfactory completion of user acceptance testing, the Project Board should sign off a System Acceptance Document to signify that the development process has been completed, and hand over all the items that will comprise the operational system to the System Owner (in practice the bulk of it will pass to the computer operations and software maintenance teams). The configuration items that typically represent a completed system are listed at Annex 1.

6.6 Audit considerations

Operational testing of IT systems is a science in itself and a detailed treatment of the subject is outside the scope of this course. The following does not therefore represent an exhaustive list of the different issues that the auditor might need to consider when reviewing the various stages of testing, as these will vary with the nature and scale of the development project.

a) Has a manager(s) with adequate authority been appointed to take overall charge of the data conversion and acceptance testing programmes?

b) Has a Data Conversion Plan been drawn up?

c) Does the Data Conversion Plan :-

- describe the data conversion strategy to be followed (e.g. the procedures for reconciling differing charts of accounts; the sequence of files to be converted; the conversion timetable; keeping converted data up-to-date)?

- allocate staff to each task (the users should be fully involved) and define specific roles and responsibilities, including that of signing off successful completion of each task?
- set out the criteria for identifying and resolving problems on the quality of the existing data (e.g. undertake file interrogation to identify missing or incompatible data items in the existing system; define procedures to deal with the correction of data rejected by the new system)?
- acceptance test any bespoke software that has been developed to support the data conversion task?
- define the controls that are to give assurance that data has been transferred completely and accurately, and correctly posted (e.g. hash and control totals, and record counts; checking a sample of detailed records back to the old system; reconciling balances between the two systems)?
- implement an effective separation of roles between those involved in transferring data and those involved in verifying that it has been correctly transferred (information security should not be neglected, particularly where financial data is involved)?
- define procedures to ensure that converted data is kept up-to-date following its transfer to the new system?
- define back up and recovery procedures for the converted data on the new system (these procedures will not relate to any processing cycle so they may differ from the eventual operational procedures)?
- define how the audit trail is to be preserved after cut over; also, how archived data from the old system will be processed after de-commissioning?

d) What arrangements have been made to ensure that the system has been correctly built (installed, configured, loaded, etc) before user acceptance testing commences?

e) Has an Acceptance Test Plan been drawn up to cover all aspects of testing?

f) Will user acceptance testing sufficiently exercise the live environment? (where testing takes place in a development environment, it is unlikely that this will provide adequate assurance that the new system will run correctly in its intended environment. This applies particularly to distributed systems that comprise extensive telecommunication networks);

g) Does the Acceptance Test Plan :-

- allocate adequate resources in terms of manpower, time and equipment to acceptance testing? (A common problem in IT projects is to reduce the time available for acceptance testing in order to recover from slippage in the overall project timetable. This can easily result in the implementation of an inadequately tested system and defective system);
- allocate individual roles and responsibilities for :-

- ♦ managing the test environment? (i.e. environment design; configuration management; operation and maintenance)
- ♦ undertaking individual tests and test cycles?
- ♦ recording test result?
- ♦ analysing test results and prioritising errors?

* fully involve the end-users in the design and execution of the acceptance testing programme?
* include ancillary procedures? (e.g. clerical control checks, the Help Desk, Network Support, System Administration);
* require the manager in charge to sign off individual tests and test cycles on successful completion?

h) Has a Configuration Manager been appointed to perform such tasks as :-

* keeping test plans, documents and test suites in step with software versions?
* providing management reports on the status of items under test?
* ensuring that change management procedures are observed?

i) Is there an adequate separation of roles to help guard against unauthorised changes taking place during testing and error correction? (e.g. between individuals involved in building and modifying configuration items; those involved in testing them; and those involved in releasing them into live use);

j) Are there adequate access controls in place to prevent unauthorised changes being made to configuration items during testing and error correction?

k) Have test data been prepared for each test? Have the anticipated results for each test been fully defined?

l) Do tests cover events that ought not to happen, as well as those that should? (e.g. do they include out of range tests; tests on processing acceptable items occurring in unacceptable combinations; duplicate transaction processing; incomplete master and standing data files);

m) Does user the Acceptance Testing Plan cover all aspects of the User Requirements Specification?

n) Are changes to defective configuration items managed in accordance with the project's change management procedures?

o) Is an adequate audit trail of changes maintained? (is it possible to back-track on a change to see how it occurred and whether it was correctly authorised?)

p) Are regression tests carried out to ensure that previously accepted areas of the new system continue to work after significant changes have been implemented?

q) Has the acceptance testing programme been signed off by the Project Board on successful completion? If not, is appropriate remedial action being taken?

Annex 1 : checklist of configuration items to be handed over on completion of acceptance testing

The following items typically represent the items handed over at the completion of a system development project. Items (a) to (n) should be handed over on completion of development and commencement of operational testing; the remainder are handed over at the completion of operational testing :-

(a) a definitive list of all configuration items handed over.

(b) software requirements and specification documents.

(c) software design documents;

(d) source code files and accompanying documentation;

(e) object code and executable files;

(f) job control libraries;

(g) confirmation of successful completion of development tests;

(h) specification of allowable system configurations;

(i) software build and installation instructions, and associated installation routines;

(j) user manuals;

(k) manuals for use by computer operations staff;

(l) technical support manuals;

(m) regression test sets;

(n) records of all errors found during development test runs and their associated change history;

(o) confirmation of successful completion of system tests;

(p) ditto of installation tests;

(q) ditto of acceptance tests;

(r) all test plans, specifications, designs and test suites used during operational testing;

(s) records of all errors found during operational testing and their associated change history.

7. System implementation

7.1 Introduction

The System Implementation stage of the SDLC comprises a number of activities the completion of which should result in an operational system. These activities do not necessarily take place in a set sequence; for example, where the system is to be implemented in stages, training and installation testing may continue for some considerable time after the new system commences live operation.

The following sections describe the activities, which, depending on circumstances, need to be completed during the Implementation Stage.

7.2 System Ownership and Administration

Although the concept of system ownership should be applied in a meaningful way in all operational financial information systems, the manner in which it is applied will differ according to circumstances. The following guidelines can therefore only assist the auditor to understand the principle and the main functions associated with it.

Each financial information system should have a recognised "owner". That is, an individual who has overall responsibility for the day-to-day security of the system and its data, and who is accountable to the Chief Executive on these issues, and also on the system's on-going usefulness to the organisation.

An important feature of system ownership is the right to authorise others to use, delete/remove, create and modify/alter system resources. System resources include data, system configuration parameters, software and documentation; also, any equipment that comes within the scope of ownership such as computers, cabling (both power and communications), environmental support and associated security devices.

In practice ownership can sometimes be difficult to attribute to an individual. This is particularly the case where different systems share common system components, such as a communications network, operating system or a database. Ownership of such infrastructure components is probably best placed with a senior member of the IT department. Despite this difficulty, senior management should agree a policy on the ownership of system resources which, when combined with sound standards for information and configuration management, ensures that all system resources have a recognised owner.

The role of System Owner is an important aspect of an organisation's system of internal control. For this reason, the System Owner must have a sound understanding of business needs and risks and of the legitimate information processing needs of each individual or functional group that is to use the system. The role should therefore be undertaken by a senior end-user of the system.

Although the System Owner must remain accountable for overall the security and on-going usefulness of a system, it may be necessary for to delegate some responsibility for day-to-day implementation, monitoring and review to a System Administrator (Annex 1); and where a system is geographically dispersed, some aspects of the Administrator role may need to be shared among local managers.

The appointment of system owners should be an aspect of an organisation's corporate information security policy. A business standard will need to complement the policy to define the system owner role; accountability; and the post holders that are to act as system owners for which systems.

7.3 Training

Training must address both the needs of the staff who are to use the system, and also those who will be responsible for operating and maintaining it. Within these two broad categories, training should also address the needs of managers whose use of the system may differ from that of their staff; for example they may need to provide electronic authorisations; obtain ad hoc reports to assist with business planning and administration, etc.

Staff should not be trained too early, or there is a risk that they will have forgotten their skills before cut-over to the new system takes place; also, staff may leave or transfer to other parts of the organisation during the intervening period. Ideally, staff should be able to work with the new system as soon as they have completed their training, and involving them in system testing and in parallel operation will help to maintain and develop their skills.

Where users are geographically dispersed, training might need be done on a "cascade" principle. This approach involves one or two of the more apt users from each location being selected for intensive training. On completion, they are then given responsibility for training the others in their work area.

It is important to monitor the quality of training as it takes place as it can easily be of limited value. For instance, where packaged systems are procured, the Operational Requirement often stipulates that the supplier provides training. Although suppliers generally know their product, they cannot be expected to know how it will fit into a particular organisation's business environment, or have detailed knowledge of the supporting clerical procedures. The project team may need to deliver additional training to address this deficiency.

During training, the students' views of the system should be canvassed. Even at this late stage it may be possible to identify potential problems or deficiencies which may be easily cured or rectified.

7.4 User groups

User Groups are useful for feeding back to management, and the System Owner in particular, solutions to problems and ideas for developing and enhancing the system, and generally making it more useful to the organisation.

A User Group should ideally be chaired by the System Owner. It should represent the various business areas covered by the system, generally at junior management level. Its aim is to meet periodically to discuss problems presented by the new system, consider how these may best be resolved, and to discuss ways in which the system might usefully be improved or enhanced. The Group should also provide a forum for

exchanging ideas on how to bypass particular problems until a permanent solution can be provided in a future release of the system (temporary solutions of this sort are sometimes referred to as "workarounds").

A system analyst might also attend meetings to provide a technical perspective on ideas, and to help in formulating system change proposals.

7.5 Service Level Agreements ("SLAs")

Where a system is managed or maintained on behalf of users by a separate service "provider" (providers could include the organisation's IT Department, a facilities management contractor, an external bureau, a telecommunications supplier, or a hardware maintenance contractor), users and providers should formally agree the standards of service to be provided, and the levels of user demand to be satisfied, before the system is implemented.

An SLA is a written agreement between the customer of an IT service and the service provider. It does not constitute a legal contract in itself, but is an essential component of it. Without an SLA there is a risk that system performance will not be related to any measure of service that users understand or require. An SLA should therefore define the required performance of the system in terms of its availability to users, response times, numbers of transactions processed, and any other appropriate criteria meaningful to the user. Performance indicators will need to be agreed, and the delivered level of service should be regularly monitored against that specified.

An SLA should also define the level of technical support to be provided to users (for example in training, help desks), the procedures for proposing changes to the system, standards of security provision and administration (e.g. system and data access controls; monitoring system and network use); contingency requirements; and a schedule of charges for the services to be provided.

Work on defining the SLA should have been started during the System Design Stage when the shape of the eventual operational system began to emerge. During the Implementation Stage, the SLA will need to be completed in detail, and signed off by the System Owner. An outline description of a typical SLA is at Annex 1.

7.6 System security policy

If the organisation is making significant use of computerised information systems, they should seriously consider formulating a Corporate Information Security Policy and, depending on circumstances, more detailed information security policies covering the particular needs of individual systems, which are critical to the organisation (financial information systems often come within this category).

The aim of an Information Security Policy is to provide management direction and support for information security.

A Corporate Information Security Policy should be endorsed by the Chief Executive and set out the fundamental security requirement for all the organisation's information systems. It should contain at least the following guidance:-

- a definition of information security, its overall objectives and scope, and its importance as an enabling mechanism for information sharing;
- a statement of the Chief Executive's intention, supporting the goals and principles of information security;
- an explanation of corporate security policies, principles, standards and compliance requirements, including:-
    - compliance with legislative and contractual requirements;
    - security education requirements;
    - computer virus prevention policy;
    - business continuity planning policy;
    - a definition of general management responsibilities and specific organisation responsibilities for all aspects of information security;
    - an explanation of the process for reporting and investigating suspected security incidents;
    - a defined review process, including frequency and review responsibilities, for maintaining the Information Security Policy.

A System Security Policy follows the same general format as the corporate security policy described above, but is focused on the particular security requirements of the system concerned. It is based on a formal risk assessment, using CRAMM or another reputable methodology, which is undertaken in parallel with the system development process (security needs were first considered in formulating the User Requirements Specification). The System Security Policy will need to be supported by detailed security mechanisms and procedures - some examples of the topics that these need to address are listed at Annex 2.

7.7 Business continuity planning

An important aspect of the System Security Policy is a requirement to produce a Business Continuity Plan that is both workable and consistent with the system's criticality to the organisation.

Most organisations now rely on computer systems; some to the extent that if their key systems are unavailable for a matter of hours, their businesses would be severely effected. It is generally impossible to fall back on manual methods, because either there is insufficient staff to process the volume of work, or staff no longer have the necessary skills; or perhaps both.

From the external auditor's perspective, an organisation's ability to maintain complete and accurate accounting records is a pre-requisite for an unqualified audit opinion. When a key financial system becomes unavailable, there is an increasing risk as time passes that the organisation will lose financial control if effective alternative arrangements for processing their financial data cannot be brought into use.

Although in practice it is often forgotten or put to one side, business continuity planning should be addressed during the SDLC, and the new system should go live with a workable plan in place to cover various failure and disaster scenarios.

The Business Continuity Plan will need to take into account the relative importance of individual systems to the organisation - this should be established using risk assessment techniques - and against this background plan how alternative computing facilities are to be provided. Following a system breakdown or disaster, highly critical systems will need to be restored first, perhaps in a matter of hours, whereas the time-span available for restoring non-essential systems may run to weeks. The maximum recovery time available for each system will determine the type and cost of the standby arrangements that are necessary.

The options for standby facilities are:-

- "do nothing"; a cost-effective option for unimportant systems which can easily be replaced "off-the shelf" (e.g. standalone PCs);
- clerical backup procedures may be feasible for systems which process low volumes of data;
- reciprocal arrangements. If made with other organisation, may be ineffective due to informality and lack of a legal foundation;
- "fortress" approach - building such a high level of protection and resilience into a system that it is unlikely to be affected by major breakdowns or disaster;
- "cold start". Suitable accommodation with power, environmental equipment, and telecommunications connections installed, but empty in all other respects. Can be either fixed or mobile;
- "hot start". Fully equipped computer room, either provided by the organisation or a contractor, and on either fixed or mobile basis;
- "mirrored" systems, which cover the most critical applications of all. These are geographically separated storage systems, which are updated in parallel, with arrangements to switch processors on command.

Other aspects of continuity planning which should also be addressed during development are the means whereby system users can also be re-located and connected to the stand-by system if necessary. The need for alternative voice and data communications, for copies of important paper documents to be stored off site (e.g. user manuals, important contracts), and for procedures for clerical processing financial data until the stand-by system is available, are also often over-looked

7.8 Cut-over

"Cut-over" is the process of transferring the live workload from the existing to the new system. The most obvious way of achieving this is simply to transfer operations, a strategy sometimes referred to as "big bang". However, while big bang may be an acceptable strategy for centralised, non-critical applications, it may be unsuitable in other situations.

First, big bang involves a high risk that significant unanticipated problems which arise following cut-over will affect business operations. Although thorough testing ought to reduce this risk to an acceptable level, it is impossible to assess fully the new

system's ability to function in an acceptable manner under simulated conditions. For this reason it is important that where a big bang approach is taken, plans are in place to revert to the previous system should the need arise during the period immediately following cut-over.

Second, the resources required to implement a big bang approach in a de-centralised system (for example, one with many local offices throughout the country) will probably be excessive, even if this approach is desirable. The implications are that each location will need to be fully installed, tested and trained on the cut-over date.

The following are some approaches to cutover that, singly or together, help to address these problems.

7.9 Phased implementation

As its name implies, phased implementation involves implementing (or "rolling out") the new system in a number of discrete steps. The approach may involve a creeping process of installation and implementation; it might involve introducing discrete system functions in a phased manner; or perhaps involve a combination of the two.

Phased implementation allows the workload to be transferred to the new system gradually. Thus, any unanticipated problems will be confined to the area(s) already implemented, and the project team have the opportunity to suspend implementation until they have been remedied. If the system is de-centralised, a phased implementation approach also allows training and installation resources to be used more effectively.

The main disadvantage of phased implementation is that it is only suitable for a system that can be usefully broken down into discrete functional and/or geographic components. It may also take a long time to complete and during this period, areas of the business are going to offer the customer varying levels of service depending on which have access to the new system and which do not.

7.10 Pilot project

This approach shares many of the features of phased implementation. It involves developing a prototype system and bringing it into limited business use.

The advantage of a pilot project is that it allows management to study the effects that a new system - or possibly a number of different options - is likely to have on the business before making a final choice. This is done without committing significant expenditure, or subjecting the business to significant risk.

The disadvantages of a pilot project are that it:-

- adds to expense, particularly if the pilot project cannot be integrated with the eventual system;
- increases the time-span before a general solution is implemented;
- may not expose problems which stem from the scale of the eventual activity;
- may differ significantly from the system as eventually installed, and might not therefore provide a reliable indicator of success.

7.11 Parallel running

Strictly speaking, parallel running is an aspect of user acceptance testing, although one which the project team will not be keen to continue for too long. It can also be considered as a cut-over strategy as the live workload is processed by the new system to confirm that it responds as desired, but before cut-over takes place. Parallel running can offer a high degree of assurance that development has been successfully completed, but without placing the organisation at risk. But this approach, while superficially attractive, has several disadvantages attached to it.

Parallel operation involves running the new and existing systems in parallel, and reconciling the outputs produced by each. It is a particularly useful way of testing an existing application that is transferred from one computer to another – for example, where a new computer replaces an obsolete model - because the same outputs will be produced on both platforms.

The advantages claimed for parallel running are that:-

- it provides a high degree of assurance that a new system will handle the operational workload in an acceptable manner;
- applications and systems software, and system tables (e.g. the chart of accounts), do not contain any errors not uncovered during simulated testing;
- it provides an opportunity for operations staff to "tune" the new system to meet actual operational conditions;
- because the existing system continues to carry the live workload during the testing period, the organisation are not at any risk from major errors or failures that might occur in the new system.

In general, where parallel running is possible, it should not be necessary to extent it beyond three processing cycles (e.g. 3 months accounts or payrolls).

In practice, project managers often avoid parallel running for one or more of the following reasons:-

- it can only be used where both systems produces broadly comparable outputs. Reconciling dissimilar reports (e.g. where the chart of accounts for the two systems differs) can be time-consuming, and maybe impossible in detail;
- it often requires additional temporary staff to be hired and trained to keep the existing system running. This adds to expense, and might result in problems with the existing system due to the temporary staff being unfamiliar with it;
- it is more suitable for testing batch processing systems, where data input and output functions are performed at one location, than on-line system where connecting the users to two different systems concurrently may be expensive and technically difficult;
- it does not test periodic processes (e.g. year-end routines) unless the period of parallel operation happens to span the period.

Some of these problems can be alleviated by "retrospective" parallel running. This involves running the new system in parallel with the existing system, but not in step.

It requires input data to be stored and processed on the new system at a convenient time, such as overnight, or at weekends or holidays.

7.12 Audit considerations

If the project is running behind schedule, there could be considerable pressure on the project team to cut back on testing and training, and this could result in significant operational problems.

The following points will need to be considered:-

a) has all relevant data been transferred to the new system in a controlled manner?

b) are backup and recovery procedures documented, and have they been tested?

c) has the training program been completed? Has any attempt been made to measure its effectiveness?

d) are user manuals clear, unambiguous and easy to understand? Do they incorporate all late changes to the system?

e) ditto documentation for work scheduling and running the system?

f) have responsibilities been assigned for carrying out clerical procedures and controls, and have they been tested?

g) has a System Administrator been appointed and trained? Are system administration activities documented?

h) has a documented plan been produced for reverting to the existing system should the need arise? Is it workable?

i) is there a system security policy? Has it been approved by the System Owner? Is it commensurate with the corporate IT Security Policy? Does it address all relevant risks? Has it been implemented?

j) has a documented business continuity plan been produced? Has it been tested? Is it workable?

k) are documented change and configuration management procedures in place?

Annex 1: Service level agreement : framework

The structure and detailed content of an SLA will depend on the working practices and requirements of the organisation concerned. However it is possible to describe a "model" set of contents as follows (NOTE! - Only items capable of being measured should be included in the SLA):

- scope of the agreement
- signatories
- date of next review, or period for which the SLA is valid
- dates of previous amendments

Note: the above 4 items must all appear on the front (title) page

- brief description of service e.g.:-
  - functions
  - applications
  - major transaction types
- service hours:-
  - hours for which service is planned to be available (if appropriate,
  - distinguishing between prime shift, maximum support, unattended periods
  - etc.)
  - special conditions for week-end, bank holidays etc.
  - relevant notes as to housekeeping, planned maintenance etc.
  - details of procedures for requesting changes to the service hours
- service availability:-
  - planned percentage availability during service hours

Note: although overall service availability is a valuable indicator, some measure of individual terminal availability, where monitoring facilities allow, may also be valuable (see Annex F for methods of expressing this)

  - maximum number of service failures to be tolerated
  - maximum amount of downtime per failure
  - maximum number of batch jobs that have to be re-run due to error, shown as a percentage of the total (jobs that have been incorrectly submitted should be identified separately)
  - measurement period (e.g. weekly, monthly, rolling 4-weekly)
  - details of any restrictions or special conditions minimum percentage of terminals available

- user support levels:-
    - Help Desk details, including procedures for reporting problems and queries, and Help Desk performance criteria
    - hours when support will be available
    - brief description of support provided
    - user guides; who will have them,
- performance:
    - target throughput rates; these should be expressed in terms meaningful to the users (e.g. business transactions rather than TP message-pairs)
    - target response time (possibly subdivided to, say, 70%, 95%, 99% of transactions)
    - target turnaround times
    - measurement period (e.g. daily, weekly, monthly, rolling 4-weekly)
- details of agreed minimum functionality (see Annex G for guidance on quantifying functionality)
- details of any service charges involved
- change control procedures:-
    - brief details of change control procedures
    - details of any planned changes that will impact on the SLA (e.g. new releases of software, new hardware)
- contingency:-
    - brief details of contingency plans in event of major failure etc., including descriptions of essential parts of the service and priorities to aid recovery following such an incident.

Note: any agreed modifications to the SLA to apply in a contingency situation (e.g. degraded service, or recovery from total loss of service) must be stipulated.

- anticipated growth : in order to assess whether service levels can be maintained for the duration of the agreement it will be necessary to have information about expected growth in business requirements; either new applications or increased transaction rates etc.
- restrictions : in order to guarantee the agreed levels of service it will be necessary to set a limit on the amount of work that can be accommodated:-
    - maximum number of transactions
    - maximum number of concurrent logged on users
    - any maximum number of registered users
- central print facilities;
    - hours available

    - printer/stationery types
    - any restrictions
- central print distribution:-
    - hours available
    - locations of distribution centres
    - description of any postal services
    - any restrictions
- user training:-
    - details of training facilities to be made available for users (courses, packages etc.) covering both equipment and services
- changes to SLA
- description of change control procedures for requesting SLA amendments.

Annex 2: IT security policy

This annex briefly describes the coverage of IT Security Policy and provides a list of suggested topics for inclusion in policy statements.

IT Security Policy

An IT security policy should exist at several levels, including:

- **national security policy**, which consist of advice and guidance promulgated by the central security authorities;
- **departmental security policy**, which covers common aspects of IT security relevant to the business;
- **business security policy**, which reflects additional requirements that are unique to a particular business environment;
- **system security policy**, which should exist for each system and cover the security rules that must be enforced.

Below is a list of topics that should be considered in the formulation of an IT security policy. Note that this is not an exhaustive list and some topics overlap the five headings.

Physical security

a) Access control to the site, building and rooms

b) Fire, flood and bomb protection

c) Disaster recovery

d) Asset management

e) Property management

f) Security furniture

Personnel security

- Security awareness programme
- Health and safety
- Staff recruitment and vetting
- The employment of contractors and consultants
- Avoidance of positive collusion
- Abuse of resources
- Education and training
- Emergency planning
- Industrial relations
- Legislation (Data Protection Act, Computer Misuse Act)
- Human error

Procedural security

- Organisation and responsibilities
- Back-up and recovery
- External transfer of information
- Business continuity planning
- Use of external software
- Software distribution
- Maintenance of equipment and software
- Security reviews and inspections
- Procurement
- Contracts management
- Control of external communications lines (e.g. dial-up)
- Repair, reuse and disposal of media and equipment
- Monitoring, tuning and reviewing security policy
- Configuration management
- Change management
- Service level agreements
- Risk analysis and management (e.g. CRAMM)
- Security incident reporting
- Authorisation of access privileges
- Auditing
- Financial management
- The overall administration of security
- Policy exemption

Document Security

- Classification and privacy markings
- Information management (e.g. data ownership)
- Disposal of consumable
- Document and media storage, transmission and disposal
- Document control

Technical Security

- Logical access controls
- Security profiles
- Audit trails

- Technical standards
- Classifying computer information
- Incorporation of security in system design and development
- Testing security features (and security accreditation)
- Evaluation (functionality, effectiveness and correctness)
- Systems architecture (e.g. distributed Vs central)
- Office systems (e.g. EMAIL, diaries)
- Open systems (e.g. directory services)
- Computer virus protection
- Innovation (use of leading edge technology)
- Formal methods (for assuring designs)
- Telecommunications and network architecture

8. Post implementation review

8.1 Introduction

A Post Implementation Review (PIR) is the final stage of a system development project. Its aim is to establish the degree of success achieved by the development project, and whether any lessons can be applied to improving the organisation's development process. In practice, a PIR is sometimes seen as an unnecessary cost (and, where a new system has failed to meet its objectives or realise benefits, it may be an uncomfortable process).

Audit considerations are addressed in the final section.

8.2 The Post Implementation Review

The full scope of a post implementation review ("PIR") will depend largely on the scale and complexity of the project. Overall, it should establish in an impartial manner whether a new system has met its:-

- business objectives (delivered within budget and deadline; is producing predicted savings and benefits, etc.)
- user expectations (user friendly, carries the workload, produces the required outputs, good response time, reliable, good ergonomics, etc.);
- technical requirements (capable of expansion, easy to operate and maintain, interfaces with other systems, low running cost, etc.).

During the PIR, it is also important to identify any lessons that can be used to improve the organisation's development process.

8.3 Timing

The timing of a PIR is much a matter of judgement. In general it should not be undertaken until after any changes and tuning that are necessary to achieve a stable system have been completed, and any significant problems have had a chance to surface. Sufficient time should also be allowed for the system's users to become familiar with it. These criteria should be met between six and twelve months after implementation. If the PIR is delayed beyond twelve months there will be an increasing risk that changing requirements - leading to further releases of the system - will obscure the outcome from the original development; also, that the need for a PIR will be overtaken by other priorities.

If there are obvious and significant problems with a new system a PIR may need to be undertaken sooner than would otherwise have been the case in order to identify the nature of the problem(s), their case(s), and to recommend a suitable course of action.

8.4 The PIR team

In order to achieve an impartial outcome, the team should be substantially independent of the original system development team. It may therefore be advisable to employ an external IS consultant to manage the review. It may also be necessary to employ other external support to assist in evaluating the delivery of technical (e.g. project management, system design) and specialised functions (e.g. in financial and

management accountancy), and to make appropriate recommendations where necessary. Internal Audit might help assess the effectiveness of internal controls.

In order to facilitate control, the PIR should have terms of reference, authorised by the approving authority, defining the:-

- scope and objectives of the review;
- criteria to be employed in measuring the achievement of objectives;
- management and organisation of the review team;
- review budget and reporting deadline.

8.5 Activities to be undertaken

During a PIR, the team should, according to their terms of reference, review:-

- the main functionality of the operational system against the User Requirements Specification;
- system performance and operation;
- the development techniques and methodologies employed;
- estimated time-scales and budgets, and identify reasons for variations;
- changes to requirements, and confirm that they were considered, authorised and implemented in accordance with change and configuration management standards;
- set out findings, conclusions and recommendations in a report for the authorising authority to consider.
- In addition to reviewing the functionality delivered by the new system, the review team will also need to look back to the Business Case on which the system was originally based to confirm that all the anticipated benefits, both tangible and intangible, have been delivered. This will involve investigating the reasons behind benefits that where not achieved, perhaps involving recommendations for remedial action, and using survey techniques to establish the extent to which intangible benefits (such as improved staff morale) have been realised.

It is also possible that the PIR will identify benefits that where not anticipated in the Business Case. These should be included in the PIR Report as additional justification for the investment, and to identify benefits that might be realised in other IS development projects.

Following their deliberations on the PIR Report, the authorising authority may either:-

- endorse continuation of the system;
- approve plans to modify the system;
- terminate the system and made arrangements for a new course of action.

8.6 Audit considerations

The following suggestions should be considered when judging the effectiveness either of a PIR, or to form the basis for the auditor to undertake one.

a) Interview business users in each functional area covered by the system, and assess their satisfaction with, and overall use of, the system.
b) Interview security, operations and maintenance staff and, within the context of their particular responsibilities, assess their reactions to the system.
c) Based on the User Requirements Specification, determine whether the system's requirements have been met. Identify the reason(s) why any requirements are not to be provided, are yet to be delivered, or which do not work properly.
d) Confirm that the previous system has been de-commissioned or establish the reason(s) why it remains in use.
e) Review system problem reports and change proposals to establish the number and nature (routine, significant, major) of problems, and changes being made to remedy them. The volume of system change activity can provide an indicator of the quality of systems development.
f) Confirm that adequate internal controls have been built into the system, that these are adequately documented, and that they are being operated correctly. Review the number and nature of internal control rejections to determine whether there are any underlying system design weaknesses.
g) Confirm that an adequate Service Level Agreement has been drawn up and implemented. Identify and report on any area where service delivery either falls below the level specified, or is inadequate in terms of what was specified.
h) Confirm that the system is being backed up in accordance with user requirements, and that it has been successfully restored from backup media.
i) Review the Business Case and determine whether:-
   - anticipate benefits have/are been achieved;
   - any unplanned benefits have been identified;
   - costs are in line with those estimated;
   - benefits and costs are falling with the anticipated time-frame.
j) Review trends in transaction throughput and growth in storage use to identify the anticipated growth of the system is in line with that forecast.

9. Rapid Application Development

9.1 Introduction

The system development technique described in these notes so far is a 'step-by-step' approach, often called the "waterfall". Although the waterfall approach is a logical and well-proven approach to system development it has a number of inherent problems associated with it, not least of which is that it is slow. Rapid Application Development, or "RAD", attempts to overcome this deficiency by taking a rather different approach to the system development life cycle.

RAD is a process for creating a system that relies substantially on controlled trial and error. It is becoming increasingly popular, particularly for the development of uncomplicated systems, because it provides a quicker development cycle than is possible using traditional methods and promises better quality systems. But to achieve success, RAD must be carefully managed; it is not an excuse for anarchy.

This chapter describes the principles of underlying, RAD and highlights the main control issues. Audit considerations are covered in the final section.

9.2 The waterfall approach

The waterfall consists of a number of discrete steps. In theory a step does not start until the previous step has been completed, although in practice this is rarely the case because some processes can be carried out in parallel (e.g. design, development and testing after overlap).

The waterfall approach places much importance on getting the business analysis and specification right before any computer programs are developed. Much time can therefore elapse before the users are able to see a working system, even in the testing stage.

Although the waterfall approach has been replaced by RAD for many types of development project, it still offers some of advantages over RAD in particular situations.

The waterfall is a more appropriate technique in situations in which the system is computationally complex, or is safety critical. This is because RAD does not aim at a complete solution, but one that delivers the main business needs in the shortest possible time. This philosophy does not fit comfortably with a requirement for a high degree of accuracy.

The main disadvantage associated with the waterfall is that it is inherently slow. It involves much analysis aimed at producing a detailed specification, the delivery of which is the ultimate goal. But a detailed specification takes time to prepare, while it is quite late in the development process (during acceptance testing) before the user is able to confirm that the specification is adequate. And where a change to the specification is needed, much work is involved in unravelling the development process in order to alter a previously accepted piece of work.

The net result is that a change in business needs that requires a new or updated computer system to be delivered quickly, cannot be met readily using the waterfall approach. This problem has long been understood, but a workable solution was not

feasible until the development of sophisticated software prototyping tools led to the emergence of RAD techniques.

9.3 What is RAD?

Although RAD is a development technique, it is not just a prescribed list of activities. It is more a philosophical approach to system development that attempts to address the problem that new systems must be built faster because software development has become a bottleneck in attaining competitive business advantage.

RAD attacks the bottleneck by making a fundamental assumption, that in practice nothing is built perfectly first time. However, 80% of the solution is often quite acceptable and can be produced in 20% of the time it would take to produce the 100% solution. RAD therefore addresses the main business needs to enable the bulk of the benefits that the new system has to offer to be delivered as quickly as possible. In RAD satisfying key business requirements has a higher priority than the quality of the system's operational characteristics - i.e. "build the right system before you build it right". Hence technical excellence, although important, becomes a means to an end rather than an end in itself.

Putting RAD into practice involves a rather different approach to the software life cycle from that taken in the waterfall. It involves :-

- concentrating on identifying and delivering the main business benefits quickly (i.e. aiming at the "80/20" solution)
- allowing project teams to take important decisions regarding the project without the need to refer back to top management for authorisation ("empowerment")
- very close co-operation between users, developers and other stake-holders (e.g the auditors) aimed at getting the right system. It is essential that sufficient trained and experienced users are attached to a development team, probably for the duration of the project
- using a life-cycle method which is designed to exploit the strengths of RAD, and also to manage its weaknesses. RAD must not become an excuse for anarchy - conformance with a sensible methodology is an essential ingredient of success
- investing in Computer Assisted Software Engineering (CASE) technology, and in training to enable developers to use it effectively and within the structure of a RAD life-cycle method.

9.4 Joint Application Development ("JAD") Workshops

JAD workshops are key features of the RAD approach to system development. They are designed to gather all project stakeholders together into one forum with the aim of reaching mutually acceptable decisions.

There are number of points in the RAD life cycle when JAD workshops should be used. These are during the :-

feasibility study to determine whether RAD is suitable approach to a particular project

definition of business requirements, where the business functions to be supported, information requirements, the users affected by the new systems and interfaces with other existing systems are identified

reviews of prototypes at various stages in the life-cycle

final review following system implementation.

The need to make important decisions means that JAD workshops should be run according to ground rules, and in a business-like manner. They are best carried out off-site at a "neutral" location where people will not be called out to attend to other responsibilities. Some pointers for the management of a JAD workshop are to:

- define the workshop objectives before the meeting, and the expected deliverables;
- choose the right mix of people and limit the workshop size to between 6 and 12 people;
- circulate pre-workshop reading material;
- allocate roles and responsibilities for the meeting;
- produce a written workshop report that details everything agreed at the meeting.

9.5 Computer Assisted Software Engineering - CASE

The term "CASE" refers to a wide range of software designed to support all stages of system development, from analysis through to implementation. CASE tools aim to improve both the rate of delivery and the quality of information systems. They achieve this by automating one or phases of the system development life cycle, such as planning, design or code generation. CASE tools have the potential to cut the resource costs of system development, and to improve the quality of the completed product.

Integrated CASE (I-CASE) are product sets that assist system development throughout the life-cycle. These product sets provide support for cross life-cycle functions, such as project management and change control.

CASE tools can improve the productivity of systems development by producing some documentation automatically, providing easy-to-learn human interfaces - especially graphics - and by providing consistency and integrity checks that identify errors at an early stage. CASE tools also improve the quality of systems by conforming to a methodology, by automating the transfer of information to and from people and other tools, and by presenting information in an easily understood format that can be amended quickly.

Not only so CASE tools require substantial investment, but they have often been used unsuccessfully. Their introduction requires careful analysis and planning, so management must be committed to, and effective in:-

- assessment of requirements
- product evaluation
- introduction of tools, or migration to them
- training and education in the use of tools
- integration of manual and computer-based tasks

- rationalisation of existing tool support.

Even though RAD depends heavily on the use of CASE tools to automate manual activities so far as possible, they are equally important in the waterfall approach. The system development activities that CASE tools can support fall under three broad headings :-

- management activities such as :-
    - project management
    - estimating
    - configuration management
    - requirements tracing (linking requirements with source code, and vice versa)
- technical activities such as :-
    - system analysis and design
    - the production of 'hand-crafted' software (editors, compilers, debuggers, testing aids)
    - rapid prototyping and simulation tools. These are designed to help analysts to define reports, screens and database requirements (users must be warned that the results they see are not a working system, but only a simulation that has been built to help them agree on the design of the final version)
    - automatic code and application generation (tools that generate COBOL, C, PASCAL, ADA etc. from design specifications. The code produced by these tools can be inefficient, and for performance reasons manual coding may be necessary)
- everything. Tools under this heading comprise Integrated Project Support Environments ("IPSEs") and Integrated CASE. They provide either a complete development environment or a set of tools that will integrate (i.e. exchange information) with each other.

9.6 Prototyping

A "prototype" is a physical working model of the proposed system (or a part of it) that is used to identify weaknesses in the users' understanding of their real requirement.

Building prototypes ("prototyping") might also be used as part of the waterfall approach. For example, to demonstrate a possible approach during the Feasibility Study or to help get the User Requirements Specification correct, after which development of the full system reverts to the waterfall. However, prototyping is a fundamental part of RAD. It is used at a number of points in the life-cycle to:-

- help broaden the users' awareness of the possibilities for a new system
- assist them in providing the developers with feedback
- provide them with confidence in (and ownership of) the new system
- preliminary research

- getting the requirements and specification correct
- helping to improve the system interface with the user ("usability")
- ensuring that the system will be able to handle full workloads successfully
- building a working model of the system.

The three main approaches to prototyping are :-

a) cowboy town - in the fashion of towns found in spaghetti westerns (i.e. facades), only the screens and reports as subsequently seen by the business are developed. There are no programs driving the screens, no database to hold data, no information layer and no linking objects defined. This type of prototype allows the development of the 'look and feel' of the system

b) pilot system - in this type of prototype one module of a much larger development is built to determine how the other modules will appear in the business

c) 'slash and burn' system - a working system is built very quickly without proper development control in order to demonstrate how it will appear to the users. Once the specification is agreed, a properly controlled development is commenced.

Prototypes often turn out to be "dead-ends". It is therefore essential that the developers are able to return to a known safe state before setting off again. Good configuration management and version control must therefore be applied to their production.

9.7 Iteration

Unlike the waterfall, the RAD approach involves only completing a step sufficiently to move onto the next. This is because RAD assumes that all previous steps in a project can be re-visited as part of an "iterative" process.

Iteration involves building a prototype and demonstrating it to the users. It can then be modified quickly (using CASE technology) to produce an amended prototype which should be closer to the users' real requirement. This process can be repeated a number of times, although not usually more than three, until users and developers are satisfied.

9.8 Object re-use

Finally, RAD makes use of "object re-use". This is the process of creating parts of software systems from existing software components rather than building them from scratch. Components may be reused without change, or they may be modified in order to reuse them.

The key benefits of object re-use are that :-

- productivity is improved through savings in design, coding and testing
- the need for skills may be reduced if complex functions are handled by re-used material.

9.9 RAD software life-cycle

It is equally important with RAD, as with the waterfall approach, that software RAD software life-cycle development is subject to a suitable life-cycle method which defines an acceptable sequence of activities and the standards (or rules) that will apply.

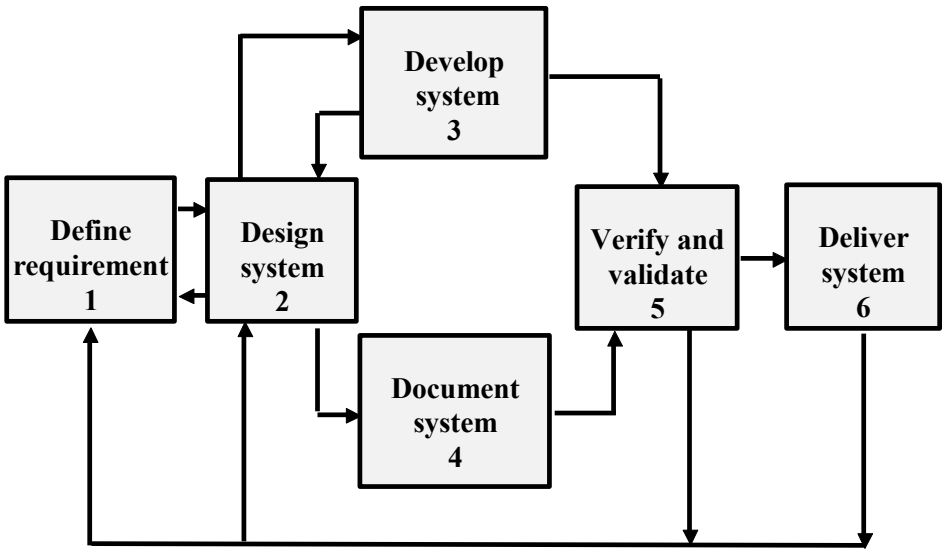


Figure 6 - RAD software development life-cycle

Following a feasibility study, the first step is to define the business requirements and to build a 'functional prototype' based on them. This will involve iteration between boxes 1, 2, 3 & 5 in figure 6.

A functional prototype aims to :-

- demonstrate the functionality and data requirements to be met by the new system;
- provide a basis for agreement with users at all levels about the direction that the project is to take.

The functional prototype should be completed and accepted by the users before further development takes place. This may involve a number of iterations, but usually not more than three (initial investigation; refinement; consolidation).

Following agreement on the functional prototype, the next stage is to design and build, and then implement the operational system (boxes 2 - 6). Here the priority is to ensure that the functional prototype is sufficiently well engineered to translate into a working system that can safely be brought into live use. This not only involves ensuring that the new system performs all the agreed functional requirements - most or all of which were demonstrated in the functional prototype - but that it also meets all the non-functional requirements as well. Non-functional requirements include those covering system performance; system back-up and recovery from failure; documentation; computer security; audit trails; multiple user access; networking facilities; operation

on a different platform; etc. It may be necessary to complete a number of prototypes leading to the system to be delivered.

In practice, the life-cycle is unlikely to be as clear-cut as figure 6 suggests. It is possible that while some components are in the design and build stage, others are just entering the functional model stage. In a small project, testing may be thinly spread throughout the project, while in a large project there may be a need to undertake a solid tranche of integration and system testing before the system enters live use. And depending on the size of the application, the dividing line between the functional model and the working system may disappear.

The actual process that is to be followed during development should be decided during the feasibility study.

9.10 Life-cycle method

RAD is not an excuse for anarchy. In order to be successful many traditional management disciplines still need to be applied. The core requirement of producing a new system by whatever method is to obtain overall good value for money. In other words, the new system should :-

- meets its defined business needs;
- meets its users' aspirations ("usability");
- be completed within cost budget and time deadline; and
- be capable of easy maintenance and enhancement.

A life-cycle method is an essential component of any information system strategy. It provides a framework within which systems may be built and maintained in a controlled manner by defining :-

- the technical and managerial policies and standards to be observed;
- a sequence of maintenance and development procedures;
- the tools and techniques to be used.

Although RAD exponents do not always agree that it is necessary, a life-cycle method should require that a Statement of Business Requirements is produced at the start of a RAD project. This is not the same as a detailed User Requirements Specification, which does not form a part of the RAD approach. It is a high level document that defines the scope and objectives of the system to be delivered and can be used to :-

- provides a benchmark against which to measure project completion
- prevent the project expanding ('snowballing') into areas beyond its originally approved boundary.

9.11 Project management

The requirements of sound project management within RAD remain the same as in the waterfall. These are to :-

- assess and manage project risks
- allocate roles and responsibilities

- draw up plans to cover the production of project deliverables (what deliverable? who will be involved? where will it be done? when will it be done? how will it be done?)
- monitor progress and ensure that the project remains on track to meet its objectives within time and budget;
- ensure effective reporting and communication, both within the team and with other interested parties.

9.12 Configuration management

Configuration management and version control also remain essential aspects of control within a RAD environment. Since there are many things happening at once and products are being delivered at a very fast rate, it is imperative that they are strictly controlled as they are completed.

Configuration management will help to ensure that :-

- developers are working on the correct components;
- developers can regress to an earlier known state if necessary (e.g. if a prototype modification is unsuccessful);
- products are protected from uncontrolled change.

The types of items that need to be brought under configuration management control should be identified at the start of the project. In general they are likely to include static models (i.e. diagrams), prototypes, important documents (e.g. statement of business requirements, the results of demonstrations and meetings), and material produced for testing purposes.

9.13 Timeboxing

Timeboxing is a control technique that helps to ensure that projects do not lose direction and run out of control. With timeboxing, the time drives the requirements, rather than the requirements driving the time as in the traditional approach to development.

Timeboxing involves setting a time by which a business deadline must be met. To be effective a time-box (which might deliver a complete part of a system or an early version of the system) should not last longer than 6 weeks, although 2 weeks being typical. A time-box must have an agreed scope and be derived from the business objectives set out in the Statement of Business Requirements.

At the end of each timebox, there should be a meeting of interested parties to present and/or demonstrate the deliverables. These meetings will also collect feedback, confirm that progress remains within user expectations, and agree on objectives for the next iteration. The time-box concept also gives top management the opportunity to abandon the project if necessary.

9.14 RAD advantages and disadvantages

Advantages

The principal advantages claimed for RAD over the waterfall approach are the ability to:-

- produce better quality systems than are possible using traditional methods

- deal with projects where the requirements are vague or unclear ("fuzzy")
- validate user requirements during the project using prototypes
- enable the users to see the system as it emerges - under the waterfall approach there is usually a long delay before the users are able to see a working system
- expenditure is not tied to the 100% solution. Instead it depends on the number of iterations that are necessary to provide an acceptable system
- in a well managed project, overall risk is reduced. This is because the early production of prototypes helps the development team to identify any significant difficulties that might lead to a project being abandoned more quickly than in the waterfall approach.

But these advantages depend on :-

- small development teams coupled with significant user/developer co-operation throughout development
- a rigorously applied life-cycle method
- the use of prototypes to study and evolve the developing system
- extensive use of CASE tools to reduce manual input and to speed up the process
- the use of time-boxes to manage the overall process.

Problems and disadvantages

RAD is essentially de-centralising in its effect. At the one extreme are the end-users who might use application generators to develop their own systems. The disadvantages here are that :-

- inattention to corporate standards for testing, documentation and security;
- problems integrating with other systems;
- unmaintainable systems;
- duplication of effort;
- diversion of resources to developments that have lower priority than other candidate systems.

Moving away from end-user developed systems, other problems that can affect RAD include :-

> Superficial systems. There is often pressure on developers to release an early, but superficial, prototype into live use. This is because the users see what they regard to be a working system at an early stage in the project, but are unable to appreciate that much work remains in order to produce an operational system of an acceptable quality. For example, a prototype might not have been designed to handle the required transaction volumes, or to provide audit trails, backup and recovery, networking facilities and adequate computer security

Snowballing. There is a temptation to add functions to the project that were not originally contemplated, and which are not essential for early delivery of a working system. This problem stems from the prototyping, evolutionary approach taken by RAD, which unless controlled can result in a system which is rich in functions, but inefficient and costly to maintain. All significant enhancements that are not present in the Statement of Business Requirements should be reviewed carefully to ensure that they are necessary.

Change control becomes more complicated with prototyped systems. Changes in design and requirements happen so quickly that they may not be documented or approved.

Poor controls in the finished system. This stems from the use of prototypes and the strong emphasis on deriving requirements based on what the user wants and sees. Security is not generally high on the users' list of priorities, while lack of a detailed systems analysis may cause the developers to overlook controls that are present in the existing system. The problem of poor security further underlines the need for a life-cycle method supported by appropriate policies and standards.

Consultant/vendor take-over. The need for experts in both the RAD technique and in the use of CASE tools often results in excessive reliance being placed on consultants, who may take over a project without transferring their skills and methods to permanent staff. As a result, the organisation may become dependent on them to maintain the operational system. Use of CASE tools (in particular IPSE and I-CASE) might result in an organisation becoming locked into one vendor's products which might not produce systems that can be used on a variety of different platforms.

Inefficiency. Automatic code generation tools do not in general provide efficient systems. If system performance is a key issue, there might be a need for handcrafted software or a more powerful hardware configuration than would otherwise be required.

9.15 Summary

The following points summarise the RAD approach :-

- it is essential that users and developers actively collaborate in RAD projects;
- development teams must be able to make important decisions without reference to higher levels of management;
- it is more important that a usable system is delivered than it is well engineered;
- systems result from a process of iterative and incremental development ('controlled trial and error');
- while there is no need for a detailed specification, there is a need for a high level Statement of Business Requirements;
- testing generally takes place throughout the development.

RAD is well suited to situations where :-

- the need is acute and the penalty for late delivery is high;
- the need is short-lived;

- the costs of control failure are low;
- the user is enthusiastic and able;
- efficiency and economy are less important than effectiveness.

9.16 Audit considerations

Auditors who are unfamiliar with RAD might find the 'trial and error' aspects of the process difficult to accept. In particular, the lack of a detailed specification in the early stage of a project prevents the auditor from being able to comment on weaknesses in user requirements, and in particular in controls. A change in audit approach will be necessary with the auditor needing to pay particular attention to records of JAD workshops, and here possible attending workshop sessions.

While the auditor should be aware of the risks that RAD brings with it, these must be balanced against the significant savings in time and cost that can result. This does not mean that RAD is an excuse for anarchy; controls are essential, but they must not stifle the benefits to be obtained from the technique. Perhaps the following reference illustrates the point.....

> "Halfway through the project, the IT Director asked for a project report. On being told that all was well - the users liked the functional prototype - he demanded a full functional specification be signed off. This took one and a half weeks to produce, and it was signed off immediately. It gathered dust while the project continued successfully, but delayed."

But regardless of the system development approach, certain fundamental aspects of good practice remain. These are the need for:-

a) a Feasibility Study and Business Case;

b) a life-cycle method to support system development and maintenance;

c) quality assurance to be applied throughout the project;

d) a formal approach to project management;

e) key measures of success against which the success of the delivered system can be judged.

Questions that the auditor will need to consider in a RAD environment will include the following :-

f) has a case been made for using RAD? (e.g. during the Feasibility Study)

g) have the development team been trained in the life-cycle method and in the use of CASE tools?

h) if consultants have been retained to support system development, have :-

- their terms of reference been defined?
- they been given targets against which to measure their achievement?
- suitable arrangements been made for them to transfer their skills to in-house staff?

i) has the project been properly initiated? (Section 2 paragraph 16)

j) is the organisation's life-cycle method and supporting standards appropriate for use with RAD? For example, does it cover :-

- project management
- prototyping;
- JAD workshops;
- use of CASE tools;
- change and configuration management;
- object re-use;
- non-functional requirements (performance, security, etc);
- quality control techniques;

k) is the life-cycle method being adhered to?

l) does the life-cycle method work in a RAD environment?

m) is there adequate user involvement in the project?

n) are JAD workshops being organised and managed properly?

o) are the reports from JAD workshops being acted on?

p) is there a Statement of Business Requirements?

q) is the project being kept within its scope and objectives set out in the Statement of Business Requirements?

r) is there effective management control to prevent the project running 'out of control'? (e.g. timeboxing; cash boxing)

s) have the minimum set of business requirements to be delivered been identified?

t) have project priorities been established that are in line with the minimum business requirements referred to in (k)?

u) are the priorities in (l) being adhered to?

v) is adequate quality control being applied throughout the project?

10. Alternative Development Methodologies

10.1 Data-Oriented System Development

Data-oriented system development (DOSD) focuses on and recognized the need for management and staff to have access to data to facilitate and support decisions. Users need and want data so they can derive information from it. Inherent in DOSD systems is the development of an accessible database of information that will provide the basis for ad hoc reporting.

The emphasis on data is not to be interpreted as the disappearance of operation-level transaction processing systems, rather it is recognition that most transaction systems have already been developed and that new systems are now addressing the users need for more information.

10.2 Object-Oriented System Development

Object-oriented system development (OOSD) is the process of solution specification and modelling. The process defines how to implement functionality defined by the analysis. OOSD is usually divided into two levels: the first, abstract design, adds solution characteristics to the analysis models but is still largely independent of development tool specifics; the second, physical design, is a more detailed level which looks at the specifics of the development environment and incorporates those features or constraint; into the designs. The major advantages of an OOSD are as follows:

- The ability to manage an unrestricted variety of data types
- Provision of a means to model complex relationships
- The capacity to meet the demands of a changing environment

10.3 Object-oriented Technology

Object-oriented technology views the raw data and the procedures governing use of data as a single object. An object-oriented approach to data management defines the object in terms of its characteristics (for example, text, graphics, format specifications and printer information) and the procedures governing their use (how those characteristics are used to make a complete document). The object is then stored as a resource that can be reused or modified. Applications that use object-oriented technology are:

- Computer-aided software engineering (CASE) for software development
- Office automation for e-mail and work orders
- Artificial intelligence (IA)
- Computer-aided manufacturing (CAM) for production and process control

The major advantages of an object-oriented approach to data management systems are:

- Ability to manage an unrestricted variety of data types
- Provides a means to model complex relationships
- Capacity to meet the demands of a changing environment

- Object-oriented models can be manipulated easily by the user
- Increased efficiency in programming through the ability to re-use elements of logic
- Ability to allow a user or an application to access only the needed information, since it stores objects independently of one another

10.4 Prototyping

Prototyping, also known as heuristic development, is the process of creating a system through controlled trial and error. It is a method, primarily using faster development tools such as 4GLs, that allows a user to see a high-level view of the workings of the proposed system within a short period.

The initial emphasis during development of the prototype is usually placed on the reports and screens, which are the system aspects most used by the end users. This allows the end user to see a working model of the proposed system within a short time.

There are two basic methods or approaches to prototyping:

- Build the model to create the design. Then, based on that model, develop the system with all the processing capabilities needed.
- Gradually build the actual system that will operate in production using a 4GL that has been determined to be appropriate for the system being built.

The problem with the first approach is that there can be considerable pressure to implement an early prototype. Often, users observing a working model cannot understand why the early prototype has to be refined further. The fact that the prototype has to be expanded to handle transaction volumes, terminal networks, back up and recovery procedures - as well as provide for auditability and control - is not often understood.

Another disadvantage of prototyping is that it often leads to functions or extras being added to the system that are not included in the initial requirements document. All major enhancements beyond the initial requirements documents should be reviewed to ensure that they meet the strategic needs of the organisation and are cost effective. Otherwise, the final system can end up being functionally rich but inefficient.

A potential risk with prototyped systems is that the finished system will have poor controls. By focusing mainly on what the user wants and what the user sees, system developers may miss some of the control that come out of the traditional system development approach, such as: backup/recovery, security and audit trails.

Change control often becomes much more complicated with prototyped systems. Changes in designs and requirements happen so quickly that they are seldom documented or approved and can escalate to a point of being unmaintainable.

Although the IS Auditor should be aware of the risks associated with prototyping, the IS Auditor should be aware that this method of systems development can provide the organization with significant time and cost savings.

10.5 Re-engineering

Re-engineering is a process of updating an existing system by extracting and reusing design and program components. This process is used to support major changes in the way an organization operates. A number of tools are now available to support this.

10.6 Reverse Engineering

Reverse engineering is the process of taking apart an application or a software application or a product, in order to see how it functions and to use that information to develop a similar system. This process can be carried out in several ways:

- Decompiling object or executable code into source code and using it to analyze the program
- Utilizing the reverse-engineered application as a black box and unveiling its functionality using test data

The major advantages of reverse engineering are:

- Faster development and reduced SDLC duration
- Creation of an improved system using the reverse-engineered application drawbacks

The IS Auditor should be aware of the following risks:

- Software license agreements often contain clauses prohibiting the license from reverse engineering the software, so that any trade secrets or programming techniques are not compromised.
- Decompilers are relatively new tools whose function depends on specific computers, operating systems and programming languages. Any change in one of these components will require developing or purchasing a new decompiler.

10.7 Structural Analysis

Structured analysis (SA) is a framework for the physical components (data and process) of an application using data flow diagrams. It includes a physical model of the system, refined data and process allocations with a revised user interface. Using SA, a review process in which a project team member leads one or more team members or customers through a segment of a deliverable he or she has produced can be implemented. Team members ask questions and make comments about technique, style, possible errors, violations or standards and other problems. In an SA process, the developer would want to do the following:

- Develop system context diagrams.
- Perform hierarchical data flow/control flow decomposition.
- Develop control transformations
- Develop mini-specifications
- Develop data dictionaries.
- Develop all external events – inputs from external environment.
- Define single transformation data flow diagrams from each external event.

The exhibit describes the strengths and weaknesses of structured analysis:

EXHIBIT

Structured Analysis Strengths & Weaknesses	
Strengths	**Weaknesses**
• Addresses user concerns quickly • Supports CASE tools • More applicable to problem-oriented analysis than design	• Weak solution representation: structured chart notation inadequate to capture concurrent task elements and their interactions (synchronous or asynchronous communication) • Does not address the issue of structuring system into concurrent tasks.

11. The external auditor's role and peer review

11.1 Introduction

For the purpose of this chapter, "external audit" is an examination of an organisation's financial accounts to ensure that they represent a true picture of the financial position. It covers the external auditor's role in relation to clients' financial systems that are either being acquired from suppliers or developed in-house, and describes the minimum involvement in the development process that is necessary to support a defensible and efficient audit of an organisation's financial statements.

User requirements are many times cheaper to design into a new system than to "bolt on" as an afterthought, assuming that it is technically feasible to do so. The external auditor should therefore be aware of the client's plans to develop or acquire new financial systems well in advance in order to confirm at an appropriate stage that essential auditing requirements will be included in the User Requirement Specification. In general the planned system :-

- must include an adequate financial audit trail (not to be confused with the 'security' audit trail which records system use - see below);
- might need to provide facilities to enable data to be selected and transferred to a microcomputer for processing using computer assisted audit techniques;
- should provide a good standard of computer security. While sound security is not generally an essential auditing requirement, it is highly desirable in order to reduce the risks of undetected errors and irregularities.

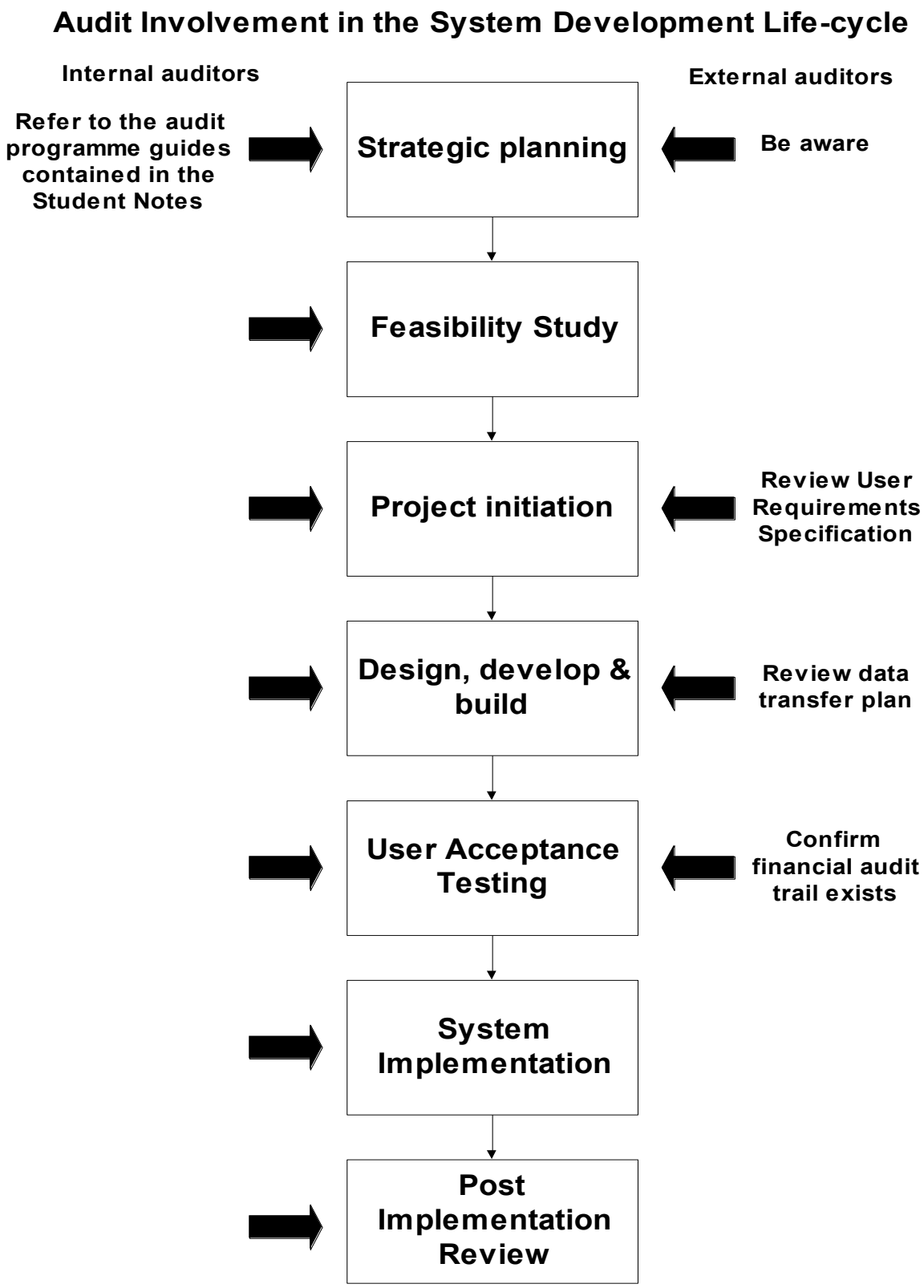
Although the external auditor's requirements of the new system do not differ from those of management, the auditor should confirm that they have not been over-looked. The auditor will also need to review the adequacy of the arrangements for transferring data between the existing and the new systems where this is to take place.

11.2 Respective responsibilities

It is for the audited organisation to determine whether the proposed development is likely to result in a system that fully satisfies business requirements. These will include the need to provide adequate internal financial controls and to produce an auditable account. In these respects, the organisation's internal audit team should advise on system specification and design.

While the external auditor may comment on aspects of the User Requirements Specification that might lead to difficulty in providing an opinion on the organisation's financial statements, it is not the external auditor's role to endorse the specification, either fully or in part. To provide such endorsement could easily place the external auditor in the position of having to report on the impact of audit advice and decisions, and on technical activities that lie well beyond the auditor's professional competence. This would clearly compromise audit objectivity.

The remainder of this chapter discusses auditing requirements and also the points in the system development life-cycle at which the external auditor should be involved



11.3 Financial audit trail

A financial audit trail is an essential audit requirement. It shows that all financial transactions in the system have been included in the relevant totals and sub-totals in the accounts; and equally that the totals and sub-totals can be shown to be the sum of

all relevant transactions. If the financial audit trail is missing or is incomplete, it may not be possible to form an unqualified opinion on a set of financial statements.

The operational system must therefore :-

- permit all financial transactions to be traced between original input, system generated transactions (e.g. EDI transactions; depreciation on assets; interest on debts) and the financial statement, in both directions, and through successive levels of summarisation;
- retain all relevant data and financial audit trail information for a sufficient time to complete the audit;
- offer a means by which the relevant computer data can be identified and captured for audit testing.

The auditor will usually be required to gather substantive evidence by selecting a number of transactions and examining them in detail to ascertain whether the underlying records meet certain auditing objectives or assertions (e.g. completeness, valuation, regularity). The audit trail will therefore need to include such information as the accounts debited and credited, the date, the operator, and references to the type and source of the transaction.

Bulk postings transferred from feeder systems (e.g. monthly payroll totals; the totals from periodic bill payment runs) must also be capable of analysis so that the underlying records can be identified, and a sample of them traced back to source documents. In addition, where system generated postings occur (e.g. depreciation, interest on debts, and stock re-ordering are sometimes automatically calculated and posted by the application software), these too must be capable of analysis and verification.

11.4 Data retention

The audit of a client's financial statements is not normally completed until some months after the end of the accounting period. It is therefore important that the new financial system will retain the individual records that comprise the audited account, together with the associated audit trail, at least until the audit has been completed.

Source documents (the manuals records that gave rise to the transaction) form an integral part of the financial audit trail, and they must be retained until the audit is complete.

Note: other statutory organisations may require information to be stored for much longer periods, for example in connection with the payment of taxes and for the preservation of records of public interest.

Where a new financial system has been introduced during a financial period, it might be necessary to provide a means whereby individual transactions can be traced between the old and new systems.

11.5 Data retrieval

If the auditor plans to use computer assisted audit techniques it will first be necessary to :-

- identify the data required for audit testing;

- capture it;
- transfer it to a micro-computer; and
- confirm that all the required data has in fact been captured.

Technical documentation is therefore necessary to enable the auditor to identify the file(s) or database(s) that holds the required data and to specify exactly which data items are needed for testing. System documentation will also need to provide information about the form in which each is data item stored (e.g. signed or unsigned binary, character, packed decimal notations) and its field size (e.g. single or double integer) so that it can be correctly imported into the audit or PC package.

The new system will also need to offer a flexible means of retrieving data. Data retrieval software (sometimes referred to as a "report writer") should be capable of capturing specified items of data and writing them to a magnetic file that can then be transferred to a microcomputer for audit testing. A flat, fixed length, ASCII format file, with comma delimited fields, is generally the easiest format to import data into specialist audit packages such as IDEA.

Data retrieval software (for example System Query Language - "SQL") is now quite widely available. It should always be run by the client on the auditor's behalf as incorrect coding could corrupt system data, while an extensive interrogation run at the wrong time could easily degrade system performance for the organisation's business users.

11.6 Computer Assisted Audit Techniques (CAATs)

(a) Use of audit packages and PC software

Auditors now make extensive use of specialist audit packages (such as IDEA), in addition to PC spreadsheet and database packages, to enhance the quality of their work. These tools can be used to :-

- total, summarise, sort, compare and select from large volumes of data in accordance with specified criteria;
- perform calculations on the data;
- perform various statistical processing; and
- provide reports designed to meet particular auditing needs.

A pre-requisite to using CAATs is for the auditor to confirm that all relevant data has been captured from the client's system(s). This is generally achieved by comparing the balances appearing in the client's accounts with similar control totals produced by the CAAT. In practice, comparisons are not always a simple matter of totalling the data extracted from the client's system(s) and comparing the results with the trail balance. Sometimes quite complex data processing has first to be carried out before a comparison is possible. In the circumstances, the auditor will need to ensure that the accounting rules to be followed in producing control totals from, for example, a transaction history file, are fully described in the system documented.

(b) Embedded audit modules

An embedded audit module is purpose-built audit software which is designed into the financial system, and which the system is designed to branch to when either :-

- a specific event occurs during processing (e.g. a particular type of transaction is encountered; or
- at a defined point in the process.

The embedded audit module passively processes the data that is presented to it. It writes control totals and copies of selected transactions to an audit file before returning control to the main process.

Use of embedded audit modules is not a standard approach to auditing computerised accounts. They are expensive to develop and maintain, add an additional processing overhead to the system, and need to be monitored closely to ensure that they do in fact correctly process all required transactions. Their use is typically confined to situations where :-

- the audit trail is transient, and information needs to be selected for auditing purposes before the audit trail is lost;
- there is a very high volume of transactions to audit (e.g. state pensions or social security payments) and an embedded audit module is the most cost-effective way of obtaining a sample transactions for audit purposes;
- transactions are of particular audit interest need to be identified and examined as soon as they occur (e.g. credit card companies use expert systems to monitor transactions in order to identify those that deviate from the norm, and which indicate the use of a stolen card).

(c) Integrated test facility

An integrated test facility can be built into a system to enable its processing integrity to be tested during live operation. It works by allowing dummy transactions to be processed by the system during live operation. These produce test results that can be compared with previously calculated expected results.

An integrated test facility can be a useful tool for monitoring system integrity in situations where system outputs cannot be related to inputs, and which cannot therefore be reconciled against one another (e.g. in job costing systems, staff time sheets - in hours - and contract rates - in money - form input, and produce invoices to customers).

In common with embedded\audit modules, integrated test facilities need to be carefully designed to ensure that, while thoroughly exercising live processes, they do not corrupt live master records or system outputs.

11.7 Internal controls

Computer security is a management responsibility, and forms an important part of the organisation's system of internal controls.

Adequate internal controls are highly desirable in any financial system. From the external auditor's perspective, their correct operation reduces the risk that error or deliberate system misuse will result in a material mis-statement in the accounts; also, that prolonged system failure or disaster will prevent complete and reliable accounting records being maintained.

An important system development task, and one in which the organisation's internal audit team should be closely involved, is to identify the internal controls that will be

necessary to reduce identified risks to acceptable levels, and which will need to be included in the User Requirements Specification for the new system. The external auditor's role should be to confirm that the client is aware of the importance of meeting basic internal control objectives on the input, storage, processing and output of financial data, and on system back-up and recovery (these requirements are dealt with in more detail in the INTOSAI module on IT controls). In systems where large sums of money are at stake and/or high system availability is required (such as in banking), it will be necessary to carry out a detailed risk assessment to identify any areas within the new system where basic controls are insufficient and will need to be enhanced.

If an organisation does not have an internal audit team or computer security specialist, the external auditor may need to assist the client by describing internal control objectives, but not particular ways of meeting them. This should be left to the organisation's security staff and the system's designers. A list of internal control objectives for financial systems is at Annex 1.

Basic computer security requires controls to prevent unauthorised users from accessing the system; and entering, changing, deleting or retrieving data. A security audit trail is also necessary to record details of those who have accessed the system, and how they have used it, and to detect any security violations that have occurred. For the security audit trail to be fully effective, every system user must be uniquely identifiable and there must also be a quick and convenient way of analysing it, such as general purpose file interrogation software.

11.8 Data conversion

The Data Conversion Plan should aim to ensure that all data is transferred completely and accurately to the new system and, where the new chart of accounts differs from the old, as is generally the case, is correctly accounted for. The auditor will also need to confirm that an adequate financial audit trail exists and that, where necessary, it extends between the old and new systems.

11.9 Audit considerations

The auditor will need to confirm that the audit trail extends between transaction initiation to summarisation in the financial statements, exists for a sufficient length of time to complete the audit, and permits the auditor to examine the following financial assertions :-

a) existence - an asset or liability exists at a given date;

b) rights and obligations - an asset or liability properly pertains to the entity at a given date;

c) occurrence - a transaction or event took place which pertains to the entity during the relevant period;

d) completeness - there are no unrecorded assets, liabilities, transactions or events or other undisclosed items;

e) valuation - an asset or liability is recorded at an appropriate carrying value;

f) measurement - a transaction or event is recorded at the proper amount and revenue or expense is allocated to the proper period;

g) presentation and disclosure - an item is disclosed, classified and described in accordance with the applicable reporting framework;

h) regularity (for UK central government systems) - a transaction is in accordance with the legislation authorising it.

11.10 Peer review

Independent peer review is now an established discipline in most significant IT projects. It employs fellow practitioners from within an organisation and/or from across Government (sometimes extending to private sector) to perform a scrutiny of a project being undertaken elsewhere in the organisation. Reviews are carried out at key stages in a project, and are designed to provide:

- project team with the benefit of advice and guidance from fellow project practitioners and
- independent assurance that the project can progress safely to the next stage of development

.1 Sponsorship is critical to the success of peer review process, hence the importance of firmly focused project "ownership". Peer reviewers must have the clear authority of a Senior Responsible Officer of the project with whom they agree terms of reference, report their findings and recommendations, and report on any conditional qualifications for proceeding to the next stage.

.2 Peer review teams

Correct team composition is very vital for the success of peer review. The peer review teams:

- need the right mix of project/programme management and technical skills. The skills required may include strategic planning for IT, negotiating procurement contracts, managing suppliers and implementing new services
- need the right mix of background: they are drawn from as many diverse parts of the department, Government or private sector
- need legitimacy: individuals are acknowledged expert practitioners in their field

11.10 Gateway process

The Gateway process represents the UK Government's implementation of the principles of independent peer review. There was no cross-government structured review process in place before January 2001, when the Gateway process was introduced. The Office of the Government Commerce is the driving force behind the Gateway initiative. The process considers a project at critical points or "gates" in its development. There are six gates through which a project must pass during its lifecycle. The six gates are:

- Gate 0 – strategic assessment
- Gate 1 – justification of the Business case
- Gate 2 – approval of procurement method
- Gate 3 – approval of the award of contract
- Gate 4 – testing whether the project is ready to go live
- Gate 5 – identifying if the project has delivered the planned benefits

Detailed checklist has been prepared for each review stage to guide reviewers.

.3 Gateway Team leader and review teams

A "project profile model" has been prescribed to determine the level of risk of the project. The composition of the review team depends on the assessed level of risk as follows:

- for large, complex, high risk projects, the Gateway team leader is appointed on the advice of the Office of Government Commerce together with a review team that is independent of the department involved;
- for medium risk projects, an independent team leader is appointed with a team drawn from independent department staff
- for low-risk projects, departments appoint the independent team leader and reviewers from within the department

11.10 Involvement of Supreme Audi Institutions in peer review

Although "Gateway" is an audit process, the National Audit Office of UK (NAO) does not have a direct role in the process. The co-ordination work is done by the Office of Government Commerce. NAO will only audit Gateway review records in the light of subsequent developments to ascertain whether the process works effectively. However, there is an opportunity for SAIs to conduct "Gateway" like milestone audits at the end of key stages of the project independently or join the peer review team. Besides, mandate such involvement would require the availability of skilled staff in adequate number to carry such reviews. Audit at various stages may demand the deployment of large resources, which may be beyond the capacity of many SAIs. May be, the SAIs could get involved in peer reviews of select projects depending on their importance and risk. Such direct participation of SAIs presents both opportunities and risks.

.4 Opportunities

- By bringing emerging problems to top management's attention at an early stage – rather than after the event – remedial action can be taken to prevent or minimise the damage that might otherwise result.
- The process could lead to an SAI building up a body of experience in IT project management, which could be gradually spread around the government domain through the work of review teams.

.5 Risks

- The project risk is unlikely to disappear entirely even after peer review
- There will be implications on the independence and objectivity of SAI should they fail to detect factors that subsequently result in an unsuccessful or abandoned project. The implications are likely to depend on whether the review process failed because:
  - it was not operated properly: a matter that could rest with the review team if the sponsor had co-operated fully and disclosed all relevant information. This emphasises the need for the review team to have the relevant skills and experience to undertake the work, together with effective leadership;
  - the process is deficient
  - the reviewer's findings are not acted on by the project owner

- of external factors, such as political decisions, that were beyond the ability of the project team to control.

These are the risks that the SAI would need to consider before taking a more proactive approach to auditing IT projects as a part of peer review team or by conducting “milestone” audits.

Annex 1: System ownership and administration

Role and responsibilities

Typically, a System Owner should undertake the following tasks those marked with an asterisk might be delegated to a System Administrator who will need to receive enhanced training in system administration and in use of the system's report writer):-

a) maintain an up-to-date risk assessment on the system;

b) ensure that the organisation's security policies are implemented within the system;

c) maintain a diary of system ownership activities, both planned and completed;

d) consider, and where appropriate approve, system change proposals to be forwarded to the Change Review Board (see note 1);

e) identify system users' training needs, and ensure that they are satisfied;

f) arrange a support service to assist users in resolving problems in their use of the system;

g) prioritise and agree deadlines with the IT department (or vendor) for the resolution of problems and the delivery of additional features;

h) assign a unique to each system user;

i) establish a secure procedure for issuing new users with an initial password, and for responding to existing users who forget their password;

j) authorise requests to access the system, and to perform specific functions within the system (note 2);

k) ensure that individual and group security profiles implement an effective separation of roles at each stage of processing;

l) ensure that individual security profiles are kept up to date;

m) maintain an inventory of user access permissions which have been granted, and ensure that the access permissions of redundant users are promptly deleted;

n) ensure that, adequate controls exist over the validity, authorisation, completeness and accuracy of all data input streams;

o) confirm that scheduled work has run without error, and that all expected output has been correctly produced and distributed;

p) review, and investigate as necessary, all run logs from batch processes, error reports, security violation reports;

q) ensure that all periodic processes (backups, daily balancing/consistency checks, production of regular management reports, etc) are scheduled to run on the due date;

r) ensure that an effective transaction audit trail is maintained for a sufficient period for financial auditing and internal control purposes;

s) identify resources to undertake user acceptance testing of system changes;

t) identify, draw up, and record test plans for all implemented changes;

u) review test results, and authorise the live implementation of system changes following satisfactory completion of acceptance testing and quality assurance review;

v) ensure that the system is adequately documented in respect of its design (including keeping the User Requirement Specification up to date), operation and use, and that documentation is updated to reflect system changes;

w) ensure that the system is backed up in a manner consistent with business needs, that at least one complete and up-to-date backup is securely stored at a remote location together with a current copy of all system documentation;

x) in conjunction with the IT Manager, periodically confirm that the system can be recovered successfully from backup;

y) ensure that a workable contingency plans exist, and that staff are familiar with the role they should play were the plan to be invoked;

z) approve any Service Level Agreements with service providers (including any necessary security clauses).

Note 1: procedures for change management will also vary according to circumstances. But in general, change proposals are considered by a panel which represents technical, as well as end user interests, with the aim of identifying any undesirable impacts which were not envisaged in the original change proposal ("impact analysis"), prioritising changes, and sometimes referring a large scale to the Board for financial sanction.

Note 2: it is good security practice to allocate system access permissions based on legitimate information processing needs, rather than the grade or status of individual system users.

Annex 2 : computer system internal control objectives

Application system control objectives

The total package of manual and systematic controls should achieve the following objectives, in a manner consistent with business risks, for each financial application system :-

1. comprehensive and up-to-date technical and user documentation (see chapter 4);
2. control over logical access to the application system (controls will need to address the identification, authentication, authorisation and audit of each user's activities);
3. a separation of roles (see note 1);
4. complete, valid, timely and appropriately authorised input data;
5. complete and accurate conversion of input data for processing by the computer;
6. complete and accurate computer processing;
7. complete and accurate computer output;
8. secure storage of computer outputs awaiting transmission;
9. transmission of computer outputs to the correct recipients in a timely manner;
10. control of financial stationery;
11. reliable data dictionary definitions;
12. reliable databases and master files;
13. reliable standing data files.

General internal control objectives

The total package of manual and systematic controls should achieve, in a manner consistent with business risks, the following objectives within the computer environment :-

1. control over physical access to premises housing financial systems, and to system resources (terminals, servers, distribution frames, routers, file libraries, system documentation, ancillary equipment, etc);
2. control over logical access to financial system resources (system configuration files, source and object code, system commands and utility programs that can bypass access controls);
3. a separation of roles within the infrastructure systems (operating system, security system, communications systems, database management, etc), and also between these and application system development and maintenance teams (see note 1);
4. effective control over all system changes (see note 2);

5. protection against corruption or loss of data (including computer virus attack);
6. continuity of operation of key financial systems in the event of serious failure or disaster.

Administrative controls

Management procedures and policies should aim to achieve the following objectives in a manner consistent with business risks :-

1. control over the recruitment of staff and their deployment to key tasks within the organisation;
2. personnel trained in appropriate skills, and to an appropriate level;
3. up-to-date definitions of individual and group missions, roles and responsibilities;
4. adequate management checking and supervision of activities, both within and surrounding the computer system;
5. an adequate separation of roles and responsibilities between internal departments;
6. an organisation for managing computer security policy and incidents.

System implementation

The following control objectives should be achieved prior to live implementation:-

1. all relevant financial data transferred completely and accurately from the old to the new system, and correctly accounted for in the new system;
2. adequate development and operational testing;
3. formal acceptance of the new system by the System Owner.