New Zealand Security of Information Technology Publication 105

CONFIGURATION MANAGEMENT

CHAPTER 1

INTRODUCTION

General

101. Configuration management is a control activity applied to the components of an IT system throughout its life to provide assurance that system components are well defined and cannot be changed without proper justification and full knowledge of the consequences. Configuration management ensures that the hardware, software, communications services, and documentation for a system can be accurately determined at any time. A set of formal configuration documentation describes the history and current state of each component, the relationships between components, any system or component incidents that have occurred, and all changes that have been applied to the system since its implementation.

102. Computer systems, once operational, are rarely immune to change. Modifications are often required to correct the inevitable system *'bugs'* that occur. Changes may be installed to provide enhanced functionality, or to reflect adjustments in user requirements. If a change is made to a security-related system component, it may also affect the security of the system in an unforeseen way. Configuration management, therefore, provides procedures to ensure that each alteration to any system component is controlled and assessed for its impact on overall system security. Security in this context includes the integrity of the software and its ability to maintain the confidentiality and integrity of the information processed.

103. An IT system's security policy will identify whether or not a configuration management regime is required. As the security policy forms part of a system's certification documentation, configuration management baseline documentation and procedures will be reviewed during initial accreditation, and evidence of formal change review will be required during re-accreditations. Proper use of configuration management will substantially simplify any re-accreditation.

Configuration Management Objectives

104. The objectives of configuration management are to:

a. provide controls to ensure the system operates correctly throughout its life;

b. ensure that the configuration of all system components is available and accurate at all times;

c. ensure the pertinent functional and physical interfaces between systems, equipments, and software are correctly and adequately documented;

d. provide maintenance efficiency by ensuring that change proposals are adequately justified and promptly actioned; and

e. ensure that the impact of any change on system functionality, security, performance, and costs is known at the time the change is approved.

Terminology

105. Configuration management requires the establishment of a definitive set of documentation for a system, called the **Configuration Baseline** . This provides a well established reference point identifying the exact configuration of the system at a specific time, to which further system development and changes can be applied. The Baseline may be re-established whenever justified by integration of change and enhancement work into the primary configuration items. This would typically be done at a major release or milestone in the life cycle of a system.

106. Each component of the system under configuration management control is known as a **Configuration Management Item (CMI)** , and is independently documented with configuration-relevant information. All proposals for system enhancement will be documented as **System Change Requests (SCRs)** and undergo management scrutiny prior to and changes being made to a CMI.

107. A **System Trouble Report (STR)** is typically raised in the event of a system problem, and would be managed within an **Incident Reporting System** . On analysis, however, the resolution may be to incorporate changes into the system. The Incident Reporting System should then cause an SCR to be raised and passed to the **Configuration Management System** . The Incident Reporting System is discussed in more detail in Chapter 5, and may be included as a module within the Configuration Management System.

108. The baseline and change documentation is known collectively as the **Configuration Management Data Base (CMDB)** . For a manual system this should exist as a collection of folders or binders which hold the required details on each CMI, with a configuration master list of all components. All STRs and SCRs should be indexed both by STR/SCR sequential number and, where known, by the relevant component name. An automated configuration management system would maintain the documentation in its own database format.

Benefits

109. Applying configuration management to an IT system can enhance a system's cost-effectiveness and security. Changes to the system, when controlled, are less error prone and therefore less costly to complete. Early identification of the impact of a change on other components can result in better designed changes. The need for formal approval of change requests will lead to better developed and properly justified changes. The application of rigorous configuration management procedures also reduces the risk of malicious changes by making each change to the system visible and ensuring full accountability and audit.

110. Development of configuration documentation provides better control over IT assets. Troubleshooting will be simplified by having available accurate system documentation. Recovery from a disaster or loss of an asset is simplified by having a clear statement of each component and its configuration state. Reversion from a troubled upgrade to a previously stable state is also facilitated. A configuration management system may provide useful asset management information, such as component maintenance costs and licence fees, appropriate to system financial planning.

111. Formal configuration management is a requirement for system accreditation. The documentation maintained for configuration management will also provide the necessary evidence to accreditation inspectors that the security aspects of each change since the system's last accreditation review have been properly evaluated. This will substantially simplify the re-accreditation process.

Costs

112. Configuration management involves resource allocation for development of the configuration baseline documentation and maintenance of configuration change records. Some costs will be incurred in the production of forms and documentation binders, and there may be additional costs associated with any computer-based configuration management system that is purchased.

113. The application of a configuration management regime may well introduce delays, sometimes with flow-on costs, to the implementation of changes. Any impact on operational effectiveness can be minimised by provision, under strict guidelines, of emergency change procedures.

114. The costs of introducing configuration management should be minimised as much as possible. While this manual provides recommended configuration management procedures and a standard format for documentation, it would be appropriate for existing procedures and documentation to be used, where suitable, for configuration management of current departmental systems.

Automated Configuration Management Systems

115. While it may be necessary to begin with a manual configuration management system, larger IT systems are likely to require the greater efficiency an automated system can offer. Planning for the introduction of an automated system should be started early enough to allow transition before the manual system fails. In most cases, the use of an automated configuration management system from the outset will be advisable.

Configuration Management and Quality Systems

116. The processes of configuration management and quality management are similar but the objectives are quite distinct. Configuration management is used primarily to identify the configuration of the system and the change state of its components; quality management ensures that the system accurately provides its defined functionality and that procedures exist to support operation of the system.

117. Quality assurance for software systems is documented as *ISO 9000.3: Guidelines for the Application of ISO 9001 to the Development, Supply, and Maintenance of Software* . This standard incorporates configuration management procedures in Section 6: Supporting Activities. The procedures documented in this NZSIT are fully compatible with the ISO 9000.3 requirements.

Configuration Management Tasks

118. Configuration management consists of two major tasks which are described in further detail in Chapters 4 and 5. They are:

a. Component Identification; and

b. Change Control, which includes:

(i) Incident Management;

(ii) Configuration Status Accounting; and

(iii) Configuration Audit.

CHAPTER 2

CONFIGURATION MANAGEMENT ORGANISATION

Configuration Control Boards

201. The control of configuration management for a system rests with a body of qualified individuals known as the **Configuration Control Board (CCB)** . There will normally be one CCB for a group of common systems in departmental use, with some staff members possibly sitting on several of the boards. The primary function of the CCB is to approve the baseline configuration management documentation, review the impact of all SCRs, approve or decline changes, and schedule SCRs into new releases of the system. SCRs should be critically reviewed to assess their cost and impact on resource usage, system stability, and security.

202. The CCB will typically involve the following people:

a. **System Sponsor**. The system sponsor or a nominated representative should sit on the CCB to provide advice on business emphasis, priorities, and requirements. The system sponsor is normally the chairman of the CCB.

b. **System Administrator**. The system administrator should sit on the CCB to advise on operational aspects of the system.

c. **System Support**. A representative from each of the system support organisations handling hardware, software, and services, as appropriate, should be included on the CCB.

d. **Quality Assurance**. It may be useful to include a quality assurance or system audit representative on the CCB.

203. The chairman of the CCB should have the authority to co-opt for specific meetings other staff members, such as technical or accreditation advisors, as appropriate. A sample Terms of Reference for the CCB is at Annex D.

IT Management Board

204. Most departments have some form of IT Management Board tasked with the policy and standards coordination of departmental IT systems. This Board should be assigned the oversight and coordination of CCB operation, and in particular be responsible for maintaining configuration management standards throughout the department and resolving issues of resource contention between CCBs. Where software maintenance staff are used to support and maintain a number of systems, the IT Management Board may need to adjudicate on priorities for software development.

CHAPTER 3

PLANNING FOR CONFIGURATION MANAGEMENT

Introduction

301. Configuration management is not a local system issue, it is a strategic IT and security matter. The adoption of configuration management as a strategy for system control is best achieved through executive direction. Formal policy endorsement will be required to provide the CCBs with the authority required to carry out their duties, and a review process should be established and instituted to ensure that the process of configuration management is being correctly carried out.

302. Introducing configuration management will require close coordination between departmental IT and security staff. An administrative structure for configuration control will need to be established through creation of a number of CCBs (see Chapter 2). The members of these Boards will need to be selected and briefed on their duties, and IT system managers advised of their responsibility for the development of baseline documentation for existing systems. It is likely that a transition period will be required for the introduction of configuration management for existing systems, as staff will often need to develop baseline documentation while still carrying out their primary roles.

303. The implementation of a configuration management regime will require planning for the transition from ad-hoc controls to the formal configuration management methodology detailed in this publication. This planning should focus in particular on staff awareness and the level of configuration detail required.

Awareness Campaign

304. Departmental staff may not be aware of the reasons for the application of configuration management to IT systems or the benefits that can accrue from such management. An awareness campaign aimed at everyone likely to be involved with the new configuration management regime should be instituted at an early stage.

305. The campaign should involve presentations to individual staff members, group presentations, information sheets, or staff circulars. It should emphasise the Chief Executive's commitment to configuration management, identify the responsibilities of the IT Management Board, and describe the CCB structure which is to be followed.

Defining the CMI Levels

306. An important issue in configuration management planning is the selection and definition of components to include under configuration management, and the level to which configuration control is to be exercised. A

system with relatively high level CMIs is said to have *'coarse'* configuration granularity, and one in which a detailed breakdown of components is recorded as CMIs is said to have a *'fine'* configuration granularity. The granularity may vary for different parts of a system. Software may, for instance be managed through fine configuration control, whereas the system documentation may be managed through coarse configuration control.

307. It is convenient to group CMIs into similar classes. Hardware, communications equipment, firmware, operating and systems software, application software, and documentation CMI classes will normally be included under configuration management. Accommodation, environmental plant, and communications services may also be included as required.

308. The system as a whole will be recorded, at the highest level, as a single CMI. This will then be described as a series of subordinate CMIs for major subsystems, each of which could then consist of further subordinate CMIs for individual modules within the subsystem. At the lowest level of definition, a CMI is known as an *'atomic CMI'* . CMIs may be connected, as for instance a terminal and a host computer. CMIs may be shared between systems or subsystems, as for instance a general purpose software library or routine.

309. The system should be described only to the level necessary for adequate management, normally to the lowest level at which changes are applied. For most systems, the following guidelines will assist in establishing the levels to which CMIs are defined:

a. Each hardware component will usually be recorded once.

b. A hardware atomic CMI will be the smallest component which, in the normal course of events, is independently installed, replaced, or modified. If boards are replaced in a PC, for example, then the board will be the atomic CMI. If the PC is swapped out for repair, then the PC itself will be the atomic CMI.

c. A software component may be recorded as multiple CMIs, each referring to a specific version of the software.

d. A software atomic CMI will normally be the smallest unit of code that can be independently recompiled.

310. It is useful to define in advance the level of atomic CMIs even if the configuration documentation or database is not immediately established down to that level. Planning for the granularity required in the longer term may avoid significant future disruption to the configuration management system. Where an automated configuration management tool is to be used, it is advisable to ensure before purchase that it does not unduly constrain the breakdown of CMIs.

311. Choosing the right CMI level requires a balance to be struck between the amount of information available for a component and the cost of maintaining the information. When a component, typically a software module,

is used in multiple systems it is usually necessary to define CMIs down to the level of that module.

312. The amount of information stored for each CMI may vary. During planning, the information requirements for each class of CMI will need to be defined. In general, the component information that will be valuable in the management of change or control of assets should be stored, and other information omitted.

Configuration Management Reviews

313. Each CMI class should be reviewed on a regular basis to ensure that an appropriate set of information is being recorded, and to incorporate any additional information needed to support change management.

CHAPTER 4

Introduction

401. CMIs under configuration management must be uniquely identifiable and should be listed in a Configuration Master List (Annex A). The primary reference to a CMI will be by some form of identification code, often referred to as the component name, and there will normally be a number of associated configuration-relevant attributes associated with the CMI and documented in a CMI Data Sheet (Annex B).

402. CMIs will normally be coarsely defined during the definition phase of a system or project. As the system evolves, the CMIs may be subdivided into finer components, producing a hierarchical CMI structure for the system.

403. In practice, systems will often use common software modules or hardware components. Such items should be held under a common use index entry, known as the primary entry, and a system-specific CMI entered, the secondary entry, which does not contain attributes but has a reference to the primary CMI.

404. The Top Level Specification, System Security Policy, Security Plan, and Standard Operating Procedures should be placed as documents under configuration control in order to simplify ongoing re-accreditation. Other system design documentation should also be included as configuration components.

Naming Conventions

405. A standard, meaningful naming convention for CMIs should be maintained to simplify configuration management and ensure that duplications are not inadvertently made. The component naming and numbering scheme may, however, be constrained by the specific automated tool used to support configuration management.

406. The naming convention should result in codes that indicate the system to which the component belongs, the type of component, the hierarchical breakdown of the CMI, and, where multiple versions exist, a version discriminator. The recommended format for CMI codes is sss.ccc[.ccc..](vvv) where

sss = system identifier

ccc = component specifier(s)

vvv = version discriminator

407. **System Identifier.** The system is the highest level component that can be considered under configuration management. A unique system name should be defined and used as a prefix to each CMI within the system.

408. **Specifier** . The specifier is used to distinguish between components of the same type for a system, and will generally reflect a hierarchical structure of system assemblies and sub-assemblies. As many specifiers as required may be used in the format.

409. **Version.** The version may be used, if required, to reflect the specific version number which is used to identify the component. For commercially sourced components, the version should reflect the vendor version or release number. Examples of this might be r1.15a or v6.2. Most CMIs, however, will have the vendor release version coded as a configuration entry attribute rather than as part of the CMI name.

Configuration Example

410. The use of CMIs can be seen in a simple example of a local area network configuration. The system is a LAN known as DAEDALUS, and it consists of PC workstations, two of which are equipped with a P-8 encryption board. This is shown in Figure 1.

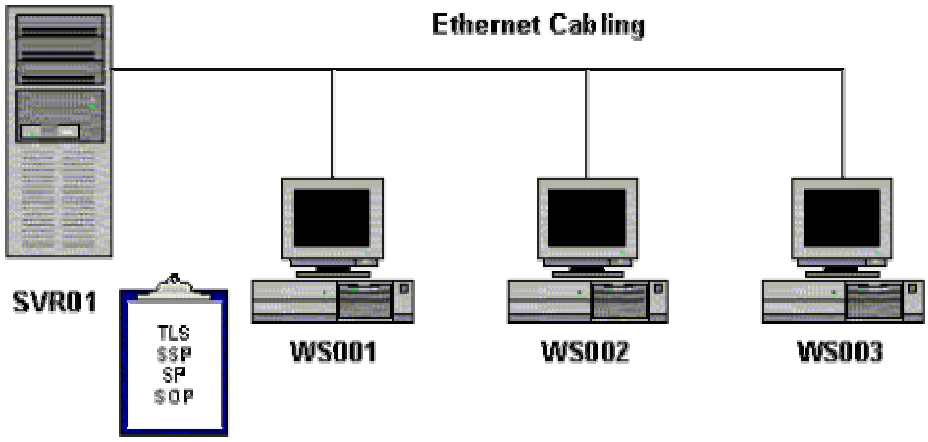

Figure 1: DAEDALUS System

411. The configuration master list for common-use items could contain entries as follows, and shown in Figure 2:

ALL.PC	Standard PC Configuration
ALL.PC.HDD	Standard Hard Disk - 170Mb
ALL.PC.MON	Standard Monitor - VGA
ALL.PC.MEM	Standard Memory - 4Mb
ALL.P8	P-8 Encryption Board
ALL.DOS	Standard OS - DOS v6.0
ALL.SLIB	Standard Software Library
ALL.SLIB.SPOOL	Standard Print Spooler

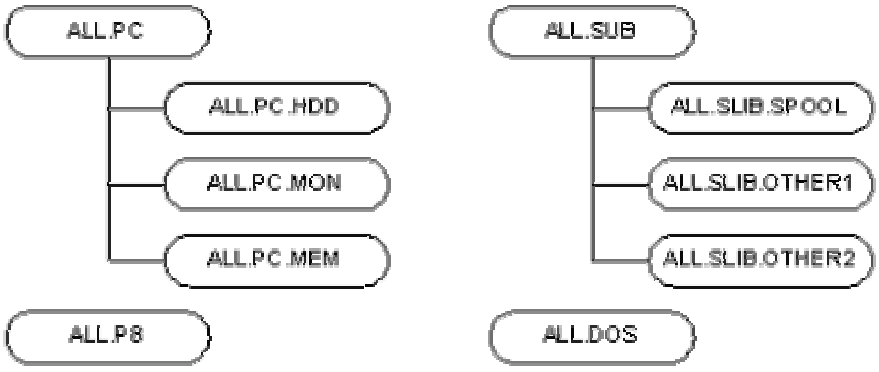

Figure 2: Common Use CMIs

412. The configuration master list for the DAEDALUS system could contain entries as follows, as shown in Figure 3:

DAEDALUS.SVR01	File server
DAEDALUS.SVR01.PC	Reference to ALL.PC
DAEDALUS.SVR01.NWARE	Netware OS
DAEDALUS.WS001	Workstation 1
DAEDALUS.WS001.PC	Reference to ALL.PC
DAEDALUS.WS001.P8	Reference to ALL.P8
DAEDALUS.WS001.DOS	Reference to ALL.DOS
DAEDALUS.WS002	Workstation 2
DAEDALUS.WS002.PC	Reference to ALL.PC
DAEDALUS.WS002.P8	Reference to ALL.P8
DAEDALUS.WS002.DOS(6.2)	DOS v6.2
DAEDALUS.WS003	Workstation 3
DAEDALUS.WS003.PC	Reference to ALL.PC
DAEDALUS.WS003.DOS	Reference to ALL.DOS
DAEDALUS.ETHER	Ethernet Cable
DAEDALUS.TLS	Top Level Specification
DAEDALUS.SSP	System Security Policy
DAEDALUS.SP	Security Plan
DAEDALUS.SOP	Standard Operating Procedures
DAEDALUS.ORCAS	The ORCAS application
DAEDALUS.ORCAS.MAIN	ORCAS main module
DAEDALUS.ORCAS.FILES	ORCAS file handling module
DAEDALUS.ORCAS.POST	ORCAS transaction posting module
DAEDALUS.ORCAS.SPOOL	Reference to ALL.SLIB.SPOOL

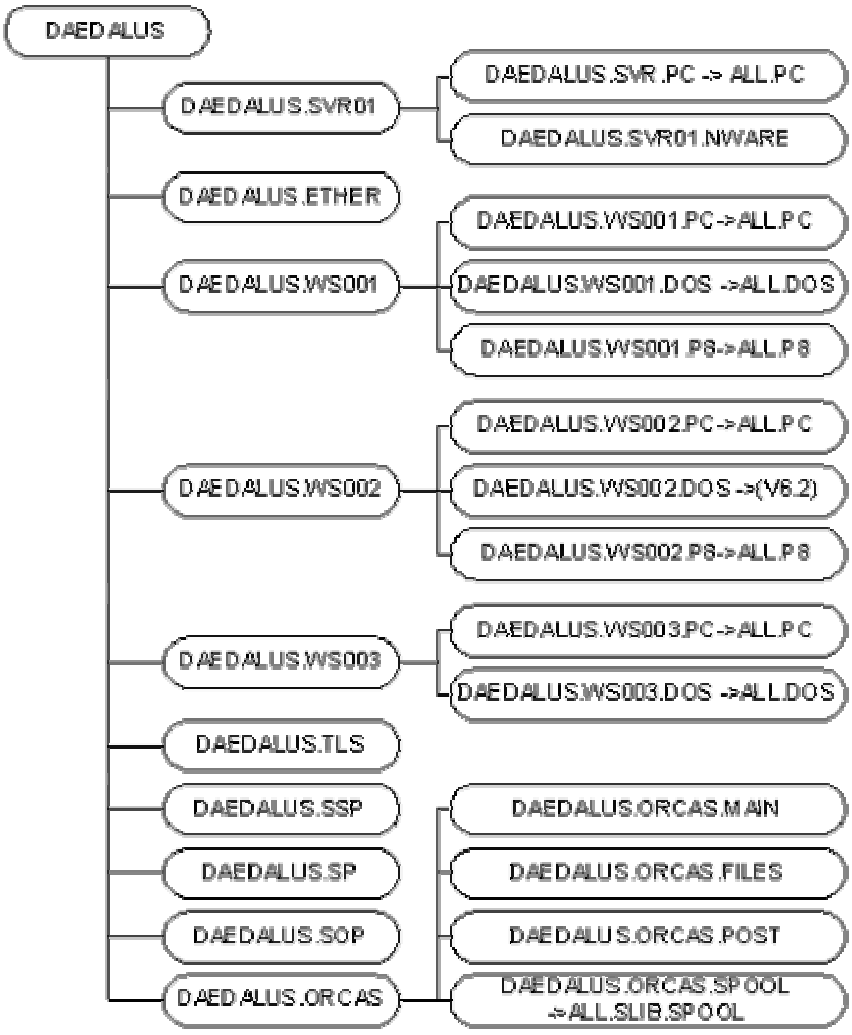

Figure 3: DAEDALUS CMIs

Component Attributes

413. There is likely to be a great deal of additional descriptive material that could be associated with each configuration entry. Such material should be selected for inclusion on the basis of its usefulness to the task of configuration management. There is little benefit including information 'for completeness' if it is never used.

414. The following attributes may be considered for inclusion in the system (not all will be relevant for each type of CMI):

a. **Model Number** . The supplier model number of the CMI.

b. **Serial Number** . The supplier or controlling authority serial number.

c. **Version Number** . The supplier version or release number.

d. **Security-Relevance**. This is a flag used to denote whether the component has any relevance to the security functionality defined in the accreditation documentation for the system.

e. **Location** The location of the CMI: a physical location in the case of documentation, firmware, or hardware components, and a reference to a software library in the case of software.

f. **Custodian**. The name and/or designation of the staff member responsible for the operational management of this CMI.

g. **Source**. The supplier details for the CMI.

h. **Purchase Date** . The date the CMI was purchased.

i. **Acceptance Date**. The date this version of the CMI was accepted into configuration management.

j. **Status**. The status of the CMI, for instance DEVELOPMENT, TEST, OPERATIONAL, ARCHIVED.

k. **Parent CMI**. For sub-assembly CMIs, the next higher CMI assembly name.

l. **Child CMIs**. For assemblies, it may be useful to explicitly record child CMIs, ie the set of components which make up this assembly. This information can, of course, be identified by scanning the CMDB for all items with a Parent CMI of this CMI name.

m. **Size and Validation Code**. The size of source and object modules should be recorded for software items, and a validation code produced through an approved hash algorithm should be recorded for each independent source, object, and executable code module.

n. **Modification Date**. The last update date for each source, object, and executable code module, and for hardware the date of last modification, should be recorded.

o. **Comments**. It is usually desirable to have some free form commentary associated with each CMI.

415. In addition to CMI attributes, the configuration management system should allow access to the SCRs for each item, in order to identify what changes have been applied to the CMI since the current configuration baseline was established.

CMI Registration

416. The first configuration baseline will normally be established after approval of baseline documentation by the Configuration Control Board (see Chapter 2) as part of system accreditation. Control of new CMIs for an existing system must be established within the configuration management system before they are delivered into the operational environment, normally as part of the item acceptance procedures. Updated versions of CMIs which are already registered in the CMDB will, of course, be registered into configuration management by amendment of their existing entry.

417. Registration procedures should ensure that software CMIs are held as operational versions of the software and not available for further development or maintenance. This is normally achieved by formal transfer of software from a development library or system to an operational library or system under operations management control.

418. Each new CMI registered into the CMDB and each update to existing items in the CMDB should create a log record in the configuration management audit trail. It may also be useful to record in the audit trail the start of development of a new version of an existing CMI, so that retrieval of component information from the CMDB will not only provide current operational information but also flag the fact that development work is proceeding.

Labelling

419. All CMIs should be labelled with their configuration master list name, and changes to configuration components should be reflected in updates to the CMI labels. Physical labels should be attached to hardware and documentation, and comment headers in source code modules should be used to label software.

Configuration Reversion

420. It may be necessary on occasions to revert from an updated configuration state back to an earlier version of the system or sub-assembly. For this reason, a full recorded history of the system configuration states is desirable to allow recovery of any viable changes made since the recovered version. This can be achieved by taking backup copies of the CMDB, by maintaining multiple CMDB versions, or through analysis and rollback from the CMDB audit log.

CHAPTER 5

CHANGE CONTROL

The Change Process

501. Changes to information systems are inevitable, and normally arise for one of two reasons. The first reason is the reported problem, which on analysis reveals deficiencies or errors in the system which necessitate a software or hardware change. The incident management process involves reporting and analysis of an incident, and may lead to a SCR being raised (Figure 4). The second reason is the user request for an enhancement to the functionality of the system.

502. The change management process involves the formal submission of change requests to the CCB for approval and scheduling into managed system releases. The change process includes controls to ensure that system, user, and configuration documentation are updated once changes have been completed and that the impact on security has been considered (Figure 5). The SCR form is commonly used in manual configuration management systems for change tracking purposes from initial request through to final change acceptance. An example of an SCR form is given at Annex C.

Incident Management

503. The configuration management system may incorporate an incident management system to record problem notification and allow tracking of problem resolution actions. Help Desk services will usually be the initial point at which problems are entered into any automated problem tracking system.

504. Problem management procedures should ensure that each time an incident, problem, or known error arises an existing problem report number is identified or a new and unique number is allocated. It may be difficult to link a problem report to any specific CMI in the early stages of problem resolution, but this link will inevitably emerge as the problem is analysed and a solution developed. The solution becomes the subject of an SCR, and may lead to one or more CMIs being updated.

SCR Details

505. While a new system release may include a number of approved SCRs, each SCR should be quite independent and a complete change in its own right. An SCR should include a number of sections:

a. **Problem Definition.** The SCR should detail the problem to be fixed or the desired change in functionality. This may relate to a reported problem already registered in the CMDB.

b. **Change Definition**. The method of incorporating the change or enhancement into the system should be detailed. If alternative approaches are feasible, the SCR should detail the reasons for selecting the proposed change.

c. **Resources**. The resources required to achieve the change should be estimated. Considerations may involve equipment or software purchase, programming time, hardware maintenance time, and other costs associated with testing and implementation.

d. **Components Affected**. The SCR should detail the components affected by the requested change.

e. **Impact**. The assessed impact of not proceeding with the change should be stated, and should consider operating cost penalties and the risk to data confidentiality, integrity, or availability should the change not proceed.

506. The CCB will balance the costs and benefits of each SCR and determine whether the change is justified.

Emergency Changes

507. It may be necessary on occasion to implement emergency changes in order to ensure a system's operational capability or integrity. Typically, and if time permits, an emergency meeting of the CCB should be held to assess the impact of applying an emergency change. Should this not be practicable, then the system sponsor or administrator should provide written authority for the change to proceed and circulate copies of the change description, with an emergency update justification, to members of the CCB out of session. Full retrospective review of the change by the CCB should take place at the earliest opportunity.

Configuration Status Accounting

508. The CCB should provide regular summary reports to the IT Management Board on the configuration status of the system under its purview. In particular, all incidents reported and changes approved for the reporting period should be detailed. Schedules and enhancement details for new system releases should be included. It is normal for configuration status accounting reports to be developed by the system administrator and submitted at CCB meetings.

Configuration Audit

509. Configuration auditing involves checking the configuration accounting information to ascertain that only authorised changes have been made to the system, and that every change made has been correctly recorded. A complete audit will involve tracing each change down through all functions to ensure that it correctly implements the requirements of the SCR.

510. Configuration audit spot checks should be carried out from time to time by the system administrator. Audits may also be carried out during scheduled and unscheduled site inspections as part of the accreditation process.

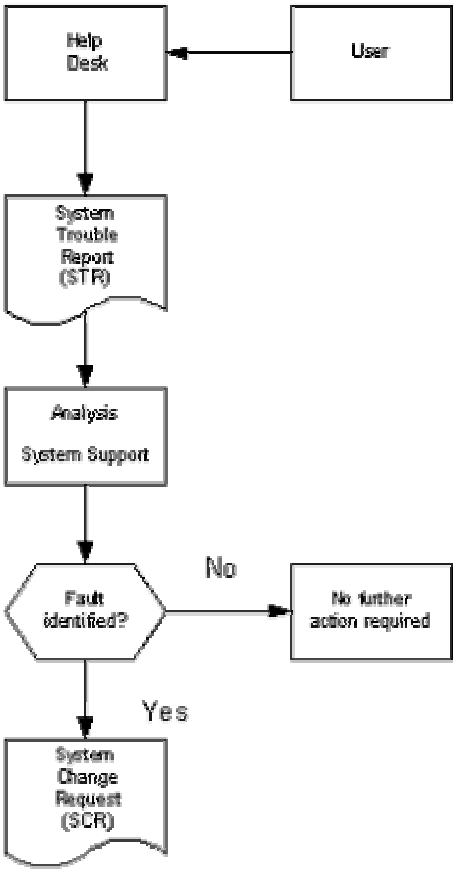

Figure 4: Incident Reporting and Analysis

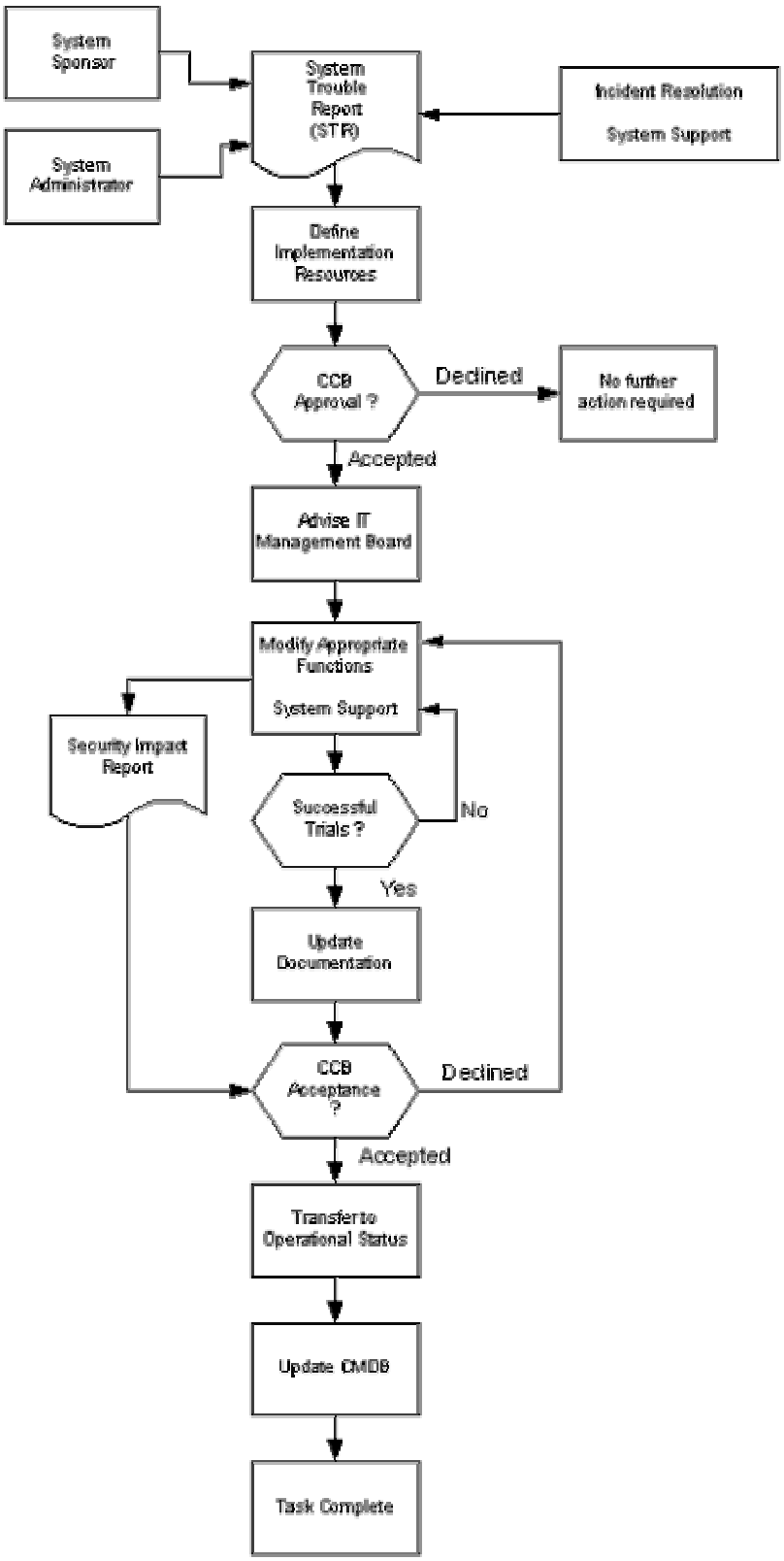

Figure 5: The Change Control Process