7.7 Service Asset and Configuration Management

Introduction

The purpose of service asset and configuration management (SACM) is to provide a logical model of the IT infrastructure. In this model the IT services are related to the different IT components needed to supply these services.

The **objective** is: to define service and infrastructure components and maintain accurate configuration records. In this context it is important that:

- The integrity of the service assets and Configuration Items (CI) are protected
- All assets and CI are located in configuration management system
- The operational and service management processes are supported effectively

Scope

All assets that are used during the Service Lifecycle fall within the scope of asset management. The process offers a complete overview of all assets, and shows who is responsible for the control and maintenance of these assets.

Configuration management ensures that all components (CI) that form part of the service or product are identified, **baselined** (the configuration) and maintained. It ensures that releases into controlled environments and operational use are on the basis of formal approvals. The process also provides a logical model of all services, assets, the physical infrastructure, and the mutual relations.

SACM also relates to non-IT assets and CI, such as work products used to develop the services and configurations items required to support the service that are not formally classified as assets. The scope of the process also includes assets and CI of other suppliers ("shared assets"), to the extent that they are relevant to the service.

Value for the business

SACM increases the visibility and performance of the service, release, or environment. Among other things this results in:

- Better forecasting and planning of changes
- Changes and releases to be assessed, planned and successfully delivered
- Incidents and problems to be resolved within the service level targets
- Better adherence to standards, legal and regulatory obligations (less non-conformances)
- Ability to identify the costs for a service

Policies

The first step is to develop and maintain the SACM policies that set the objectives, scope and principles and critical success factors for what is to be achieved by the process. There are significant costs and resource implications to implementing SACM and therefore strategic decisions need to be made about the priorities to be addressed. Many IT service providers focus initially on the basic IT assets (hardware

and software) and services that are business-critical or covered by legal and regulatory compliance e.g. SOX, software licensing.

Starting points

The policies describe the starting points for the development and control of assets and CI, for instance:

- The costs of SACM are proportionate to the potential risks to the service if SACM were not implemented
- The need to provide specifications for "corporate governance"
- The need to guarantee the agreements in the SLA and other contracts
- The specifications for available, reliable and cost-effective services
- The specifications for performance criteria
- The transition from reactive maintenance to proactive control
- The requirement to maintain adequate asset and configuration information for stakeholders

Basic concepts

Service assets, configuration items, configuration records, the CMS and the SKMS

It is important to distinguish between service assets, configuration items and configuration records, as these concepts are often confused.

Service asset

A service asset is any resource or capability that could contribute to the delivery of a service.

Examples of service assets include resources such as:

- **Funds** – in the form of budgets to run, grow and/or transform the business
- **Applications** – commercial-off-the-shelf, developed internally, purchased from vendors
- **Infrastructure** –hardware systems or components with a unique identifier
- **Information** – All platforms and all file formats used by the organisation

Examples of service assets include capabilities the ability and the (delegated) authority of management team to make decisions, the ability of the organisation to respond to situations, the ability of the processes to be as effective and as efficient as possible, and the knowledge of the IT personnel.

Note that people are a type of resource in the form of staff numbers and their availability at a given time. People are also a capability type in the form of their skills (technical and soft), knowledge, experience, and aptitudes. It is important to remember that the performance of people is influenced by many factors such as attitudes, behaviours, and cultures of the organisation.

Note: service assets and customer assets are essentially defined the same way. The primary difference is that *service assets* are used by the service provider while *customer assets* are used by the customers (within the business).

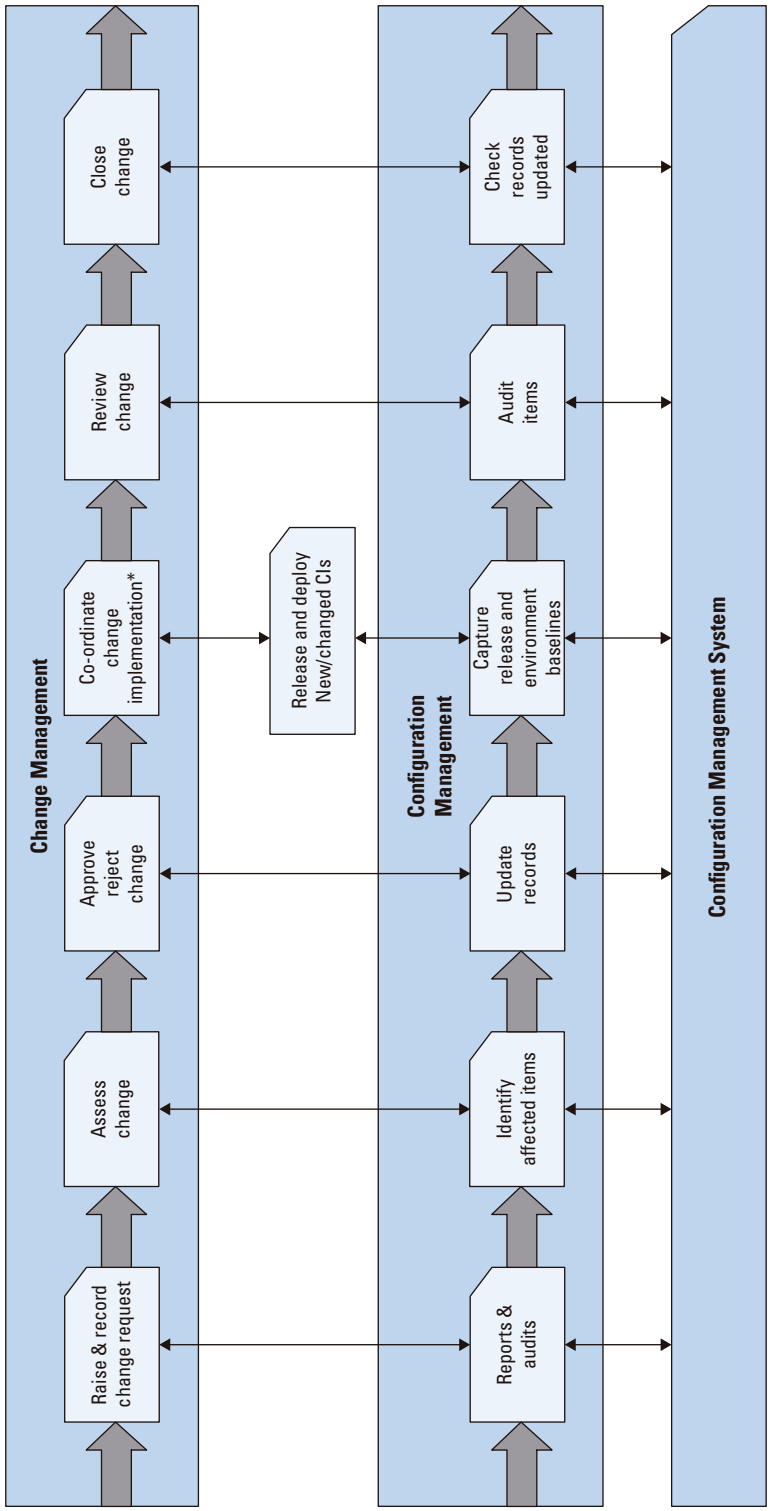

Figure 7.5 Sample change and service asset & configuration management workflow
Source: The Cabinet Office

Configuration item
A configuration item (CI) is a service asset that needs to be managed in order to deliver an IT service. Every CI is under the control of the change management process. Note that a change considered as a "pre-approved standard change" is part of the change management process even if it does not have to be submitted to the CAB.

Configuration record
A configuration record is stored in a CMDB and managed with a CMS. The CI record contains a set of attributes and relationships about a CI – either parent-child or peer-to-peer. It is important to note that a CI is not stored in a CMDB; the configuration records in the CMDB describe the CI.

The configuration management database (CMDB)
The CMDB is a set of tools and databases that are used to manage data and information about configuration items. The CMDB is a federated and logical model where multiple individual databases are linked together.

The configuration management system (CMS)
The CMS is a set of the tools, files, and databases that are used to manage the data, information, and knowledge used by the service provider to run itself and support the business. The SACM process is responsible for managing the CMS. the SACM process owns some items in the CMS but others – items such as incident, problem, known error, change, release databases, the management information systems of the service design processes (to name but a few) – will be owned and managed by respective processes.

The service knowledge management system (SKMS)
The service knowledge management system (SKMS) is a set of the tools, files, and databases delivered and supported by the service provider to manage the data, information, and knowledge used by the business to accomplish their outcomes. The SACM process is *not* responsible for managing the SKMS. As with the CMS, various items will be owned and managed by various processes. The knowledge management process is responsible for managing the SKMS.

The CMS and the SKMS generally consists of four logical layers:

Table 7.4 Logical layers of CMS and SKMS

Presentation layer (top layer)	To provide different "views" of the three previous layers to the service provider and to the business
Knowledge layer	To process the information into meaningful reports and queries for analysis purposes
Information layer	To collate, structure, and integrate the data into meaningful information
Data layer (bottom layer)	To collect the data and information from different sources in different file formats

The configuration model

Service asset and configuration management delivers a model of the services, assets, service-assets, customer-assets, and the infrastructure by recording the relationships between configuration items. Configuration models are used to:

- Assess the impact and cause of incidents and problems
- Assess the impact of proposed changes
- Plan and design new or changed services
- Plan technology refresh and software upgrades
- Plan releases
- Optimise asset utilisation and costs

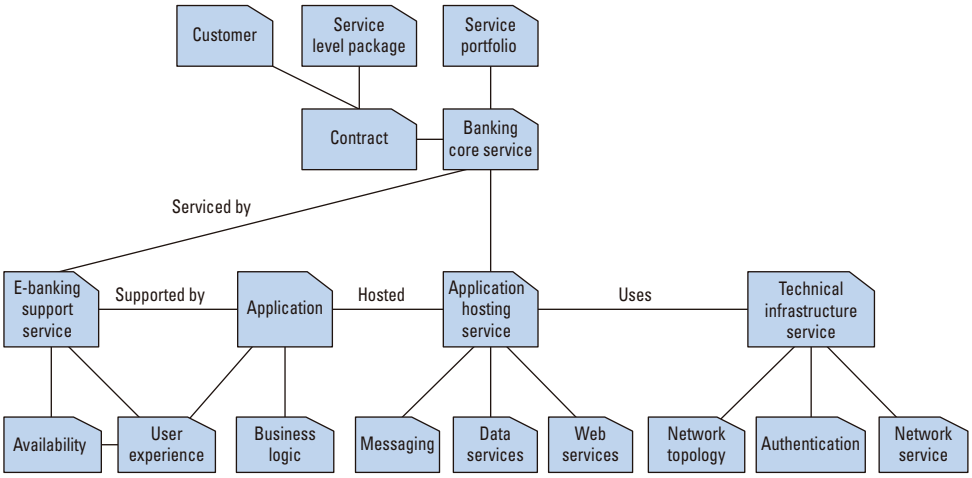

Figure 7.6 Sample logical configuration model

Source: The Cabinet Office

CI types

There are many types of CI, such as:

- Service lifecycle CI
- Service CI
- Organisational CI
- Internal CI
- External CI
- Interface CI

Where possible, automated tools (such as discovery, inventory and audit tools) are used to populate and maintain the CMDB, CMS and SKMS. This minimises the opportunity for error and saves costs.

Libraries and stores

ITIL defines various libraries to support the CMDB, the CMS, and the SKMS. The libraries will of course be referenced in the CMDB in the form of CI records. They are also considered part of the CMS or the SKMS.

Table 7.5 Libraries and stores

Definition	Example	
A **secure media library** is a collection of software, electronic or document CI of a known type and status.	Definitive Media Library (DML)	This is a secure store where the definitive, authorised (approved) versions of all media CI are stored and protected.
A **secure hardware store** is a secure location (storage space) where IT assets are stored.	Definitive spares	These are areas set aside for secure storage of definitive hardware spares.

Configuration baseline

A *configuration baseline* is a configuration of a service, or part of a service, that has been formally reviewed, agreed, and signed off. It serves as the basis for further activities, which can only be changed through formal change procedures. Configuration baselines are included in the CMS. A baseline is also used to:

- Mark a milestone in the lifecycle of a service
- Build a service component from a defined set of inputs
- Change or rebuild a specific version at a later stage
- Assemble all relevant components in readiness for a change or release
- Fall-back to a previous known state in the case of issues during or after a change

Snapshot

A *snapshot* (moment in time) is the most recent status of a CI or environment. A snapshot is stored in the CMS and is kept as a read-only historical record. A snapshot can be used to assist:

- Problem management in analysing the situation at the time incidents occurred
- Security management in facilitating system restore for security scanning software

Activities, methods, and techniques

The basic SACM process activities consist of:

- Management and planning
- Configuration identification
- Configuration control
- Status accounting and reporting
- Verification and audit

1. Management and planning

The management team and configuration management decide what level of configuration management is needed and how this level will be achieved. This is documented in a **configuration management plan**. Contents of a configuration management plan include:

- Context and objective
- Scope
- Requirements
- Applicable policies and standards

- Organisation for configuration management including roles and responsibilities
- SACM systems and tools
- Selection and application of processes and procedures to implement SACM activities including configuration identification, version management, build management, etc.
- Reference implementation plan
- Relationship management and interface controls e.g. with financial asset management, with projects, with development and customers, etc.
- Relationship management and control of service providers and sub-contractors

2. Configuration identification
Configuration identification focuses on determining and maintaining the naming and version numbering of assets and CI, the mutual relations and the relevant attributes. The most important activities of configuration identification are:
- Defining and documenting criteria for the selection of CI and the components within them
- Selecting the CI on the basis of the defined criteria
- Giving all CI unique numbers for identification purposes
- Specifying the attributes of each CI
- Indicating when each CI will be placed under configuration management
- Identifying the "owner" of each CI

Create a **configuration structure** for each IT service. This structure shows the relations and hierarchy between CI for a certain service. The configuration structure has a top-down approach. The highest level is the service in question. The lowest CI level is determined by:
- The available information
- The necessary level of control
- The necessary resources required to maintain that CI level

It is only useful to include a certain CI if it supports the other service management processes.

Document **naming conventions** and apply them in the identification of CI, documents and changes, but also, for instance, in basic configurations, releases, and compilations.

Each CI must be uniquely identifiable by means of a version number, as there may be several versions of the same CI; for example different versions of software. During the planning of the naming conventions, allow for future growth. The names must also be short but meaningful, and correspond with the existing conventions as much as possible.

Provide all physical CI, such as hardware, with **labels**, so that they are easy to identify. The labels must be easy to read and accessible, so that a user can pass the information on the label on to the service desk. Labels with a barcode are very efficient for physical

audits of the CI. During such audits it is checked whether the CI in the organisation correspond with those within the CMS.

With the aid of **attributes** information is stored that is relevant for the CI in question. Note that different CI types require different attributes. This is one of the reasons why there are many CMDBs. A common attribute must be identified and used to link the various CI types found in various CMDBs. A relationship attribute is one such way to accomplish this. The following is a list of *potential* attributes.

Table 7.6 Potential CI attributes

CI unique identifier (number/label or barcode)	Related software
CI type	Historical data (verification or audit trail)
Name	Relationship type
Description	Applicable agreements (SLA, OLA, UC)
Version	Purchase date
Location	Acceptance date
Licence details (type, expiry date)	Current status
Owner	Planned status
Status	Purchase value
Supplier/source	Residual value after depreciation
Related documents	Comments

The characteristics of a CI or CI type are often recorded in **configuration documentation**.

Table 7.7 is a *sample* RACI table (*Responsible, Accountable, Consulted, and Informed*) that shows different documentation types of service CI and indicates who takes responsibility for the documents in the different phases of the Service Lifecycle.

Relationships describe how CI work together to provide a service. The CMDB maintains these relations to demonstrate the interdependencies between CI, for instance:

- A CI **is part of** another CI – a software module is part of an application (parent-child relation)
- A CI **is connected to** – a workstation is connected to the LAN
- A CI **uses** another CI – a business application uses a database
- A CI **is installed on** another CI – word processor is installed on a PC

Relationships are also the mechanism for associating RFC, incidents, problems, known errors, and releases to CI. Relations can be 1-to-1, 1-to-many and many-to-1. Configuration items are classed by means of a **classification**, for instance: service, hardware, software documentation, personnel.

Table 7.7 RACI table for configuration documentation in the lifecycle

Service Lifecycle State	Examples of Service Lifecycle Assets and CI impacted	Service Strategy	Service Design	Service Transition	Service Operations	CSI
Service Strategy	Portfolios – service contract, customer Service Strategy requirements Service Lifecycle Model	A	C	C	R	C
Service Design	Service Package (including SLA) Service Design Package e.g. Service Model, Contract, Supplier's service management plan, Process interface definition, Customer Engagement plan Release Policy Release Package definition	I	A	C	R	C
Service Transition	Service Transition model Test plan Controlled environments Build/Installation plan Build specification Release plan Deployment plan CMS SKMS Release Package Release baseline Release documentation Test report	I	C	A	R	C
Service Operations	Service Operations model Service Support model Service Desk User assets User documentation Operations documentation Support documentation	I	C	C	A/R	R
Continual Service Improvement	CSI model Service improvement plan Service reporting process	A/C	A/C	A/C	R	A

3. Configuration control

Configuration control ensures that the CI is adequately managed. No CI can be added, modified, replaced, or removed without following the agreed procedure. Establish guidelines and procedures for, among others:

- License management
- Change management
- Version management

- Access control
- Build control
- Promotion
- Deployment
- Installation
- Baseline configurations integrity management

4. Status accounting and reporting

The lifecycle of a component is classified into different stages and the stages that different types of CI go through must be properly documented. For instance, a release goes through the following stages: registered, accepted, installed, and withdrawn.

Status reports give an insight into the current and historical data of each CI and the status changes that have occurred.

Different types of **service asset and configuration reports** are needed for configuration management. The reports may relate to individuals CI, but also to a complete service. Such reports may consist of:

- A list of CI and their baseline
- Details on the current status and change history
- A list of unauthorised CI detected
- Reports on the unauthorised use of hardware and software

5. Verification and audit

SACM conducts verifications and audits to ensure that:

- There are no discrepancies between the documented baselines and the actual business environment to which they refer
- CI physically exist in the organisation or DML and spares stores, the functional and operational characteristics of CI can be verified and checks can be made that records in the CMDB match the physical infrastructure
- Release and configuration documentation is present before the release is rolled out

Table 7.8 Verification and audit

Verification	Regular (on-going) activity responsible for ensuring that information in the CMDB is accurate and that all configuration items have been identified and recorded in the CMDB
Audit	Periodical formal inspection to check whether a standard or set of guidelines is being followed, that records are accurate, or that efficiency and effectiveness targets are being met. An audit may be carried out by internal or external groups

Document all exceptions resulting from the verifications and audits and report them. Corrective actions (CI that need to be added, changed, or deleted) are handled via the change management process.

Verification or audits are conducted at the following times:
- Shortly after changes to the CMS
- Before and after changes to IT services or infrastructure
- At random and planned intervals
- Before a release to ensure the environment is as expected
- In response to the detection of unauthorised CI
- Following recovery from disasters

Audit tools can perform checks at regular intervals, for instance weekly. For example a desktop audit tool compares the configuration of an individual's desktop against the "master" configuration that was installed.

Information management

- Backup copies of the CMS should be taken regularly and stored securely. It is advisable for one copy to be kept at a remote location for use in the event of a disaster
- The CMS contains information on backup copies of CI. It will also contain historical records of CI and CI versions that are archived, and possibly also of deleted CI or CI versions
- Typically, the CMS should contain records only for items that are physically available or could be easily created using procedures known to, and under the control of, service asset and configuration management
- The CMS includes pointers to knowledge and information assets that are stored in the SKMS, and it is important to maintain these links and to verify their validity as part of regular audits
- SACM is responsible for the maintenance of many knowledge and information assets within the SKMS, and these must be maintained with the same level of control as the CMS

Interfaces

By being the single process for managing configuration data and information for IT service management, SACM supports and interfaces with every other service management process and activity to some degree. However, some of the most visible relationships include but are not limited to:
- **Change management** – Identifying the impact of proposed changes
- **Financial management for IT services** – To capture key financial information such as cost, depreciation methods, owner and user, maintenance and repair costs
- **ITSCM** – For increased awareness of the assets on which the business services depend, control of key spares and software
- **Incident and problem management** – To provide and maintain key diagnostic information
- **Availability management** – To assist in the detection of points of failure
- **Change and release and deployment management** – To benefit from a single coordinated planning approach
- **Configuration control is synonymous with change control** – Understanding and capturing updates to the infrastructure and services

- SACM also has close relationships with some business processes, especially fixed asset management and procurement

Triggers

Here are some of the triggers that may start the whole, or part of the, service asset and configuration management process.

- Updates from change management
- Updates from release and deployment management
- Purchase orders
- Acquisitions
- Service requests

Inputs

Here are some of the inputs required by the service asset and configuration management process.

- Designs, plans and configurations from service design packages
- Requests for change and work orders from change management
- Actual configuration information collected by tools and audits
- Information in the organisation's fixed asset register

Outputs

Here are some of the outputs produced by the service asset and configuration management process.

- New and updated configuration records
- Updated asset information for use in updating the fixed asset register
- Information about attributes and relationships of configuration items
- Configuration snapshots and baselines
- Status reports and other consolidated configuration information
- Audit reports

Critical success factors

The critical success factors, which change over time, for the service asset and configuration management process may include but are not limited to the following.

- Accounting for, managing and protecting the integrity of CI throughout the service lifecycle
- Supporting efficient and effective service management processes by providing accurate configuration information at the right time
- Establishing and maintaining an accurate and complete configuration management system (CMS)

Metrics

The performance of the service asset and configuration management process can be measured according to:

- Improved accuracy regarding the assets utilised by each customer or business unit
- Increase in re-use and redistribution of under-utilised resources and assets
- Reduction in the use of unauthorised service- and customer-assets

- Reduced number of exceptions reported during configuration audits
- Reduction in time and cost of diagnosing and resolving incidents and problems
- Improved ratio of used licences against paid-for licences
- Reduction in risks due to early identification of unauthorised change
- Improved audit compliance
- Shorter audits as quality configuration information is easily accessible

Challenges

Here are some of the potential challenges faced by the service asset and configuration management process

- Persuading technical support personnel to adopt a checking in/out policy
- Attracting and justifying funding for SACM
- An attitude of "just collecting data because it is possible to do"
- Lack of commitment and support from management

Risks

Here are some of the potential risks faced by the service asset and configuration management process

- The temptation to focus on technology rather than service and business needs
- Degradation of the accuracy of configuration information over time
- Setting the scope too wide
- Setting the scope too narrow
- The CMS becomes out of date due to the movement of hardware assets by non-authorised personnel

7.8 Release and Deployment Management

Introduction

ITIL defines release and deployment management as follows:

> *"Release and deployment management aims to build, test, and deliver the capability to provide the services specified by service design and that will accomplish the stakeholders' requirements and deliver the intended objectives."*

The purpose of release and deployment management is the deployment of releases into production, and the establishment of effective use of the service in order to deliver value to the customer and to be able to hand over to service operations.

The **objective** of release and deployment management is to ensure that:

- Release and deployment plans are in place
- Release packages (compilation) are deployed successfully
- Knowledge transfer to the customers takes place
- There is minimum disruption to the services

Scope
The processes, systems, and functions for packaging, building, testing, and deployment of a release into production and establishment of the service specified in the service design package before final handover to service operations.

Value for the business
Effective release and deployment management contributes to the business because:
- Changes are realised faster, cheaper and with fewer risks, and the operational objectives are supported better
- The implementation approach is more consistent and the traceability requirements (e.g. audits, legislation etc.) are complied with more closely

Basic concepts

A **release** is a set of new or changed configuration items that are tested and will be implemented into production together.

A **release unit** is the portion of the service or infrastructure that is included in the release, in accordance with the organisation's release policy and guidelines. Releases are documented in the CMS for the support of the release and deployment process.

It is important to determine the correct level of the release. For a business critical application it may make sense to include the complete application in the release unit, but for a website it may only have to be the HTML page that is changed.

Releases can be classified into the following **release categories**:
- **Major releases** – Important deployment of new hardware and software with, in most cases, a considerable expansion of the functionality (V1.0, 2.0, etc.)
- **Minor releases** – These usually contain a number of smaller improvements; some of these improvements were previously implemented as quick fixes but are now included integrally within a release (V1.1, V1.2, etc.)
- **Emergency releases** – Usually implemented as a temporary solution for a problem or known error (V1.1.1, V1.1.2, etc.)

In the **release design** different considerations apply in respect of the way in which the release is deployed. The most frequently occurring options for the rollout of releases are:
- **Big bang versus phased** – A big bang release deploys the new or changed service for all the users at the same time. A phased deployment deploys the release for part of the user base at a time.
- **Push and pull** – With a push approach the service component is deployment from the "centre" to the target locations. With a pull approach the new release is made available to the users from a central location from which they download to their location at a time they choose.
- **Automated or manual** – Releases can, to a large extent, be automated; e.g. the use of installation software.

The release and deployment teams must have a good understanding of the relevant IT architecture involved in a release and deployment process. This understanding is essential to determine the sequence in which the activities are implemented and for mapping out all the interdependencies.

A **release package** is a single release unit or a structured set of release units. In the case of a new or a changed service all the elements of which the service consists – the infrastructure, hardware, software, applications, documentation, knowledge, etc. – must be taken into account.

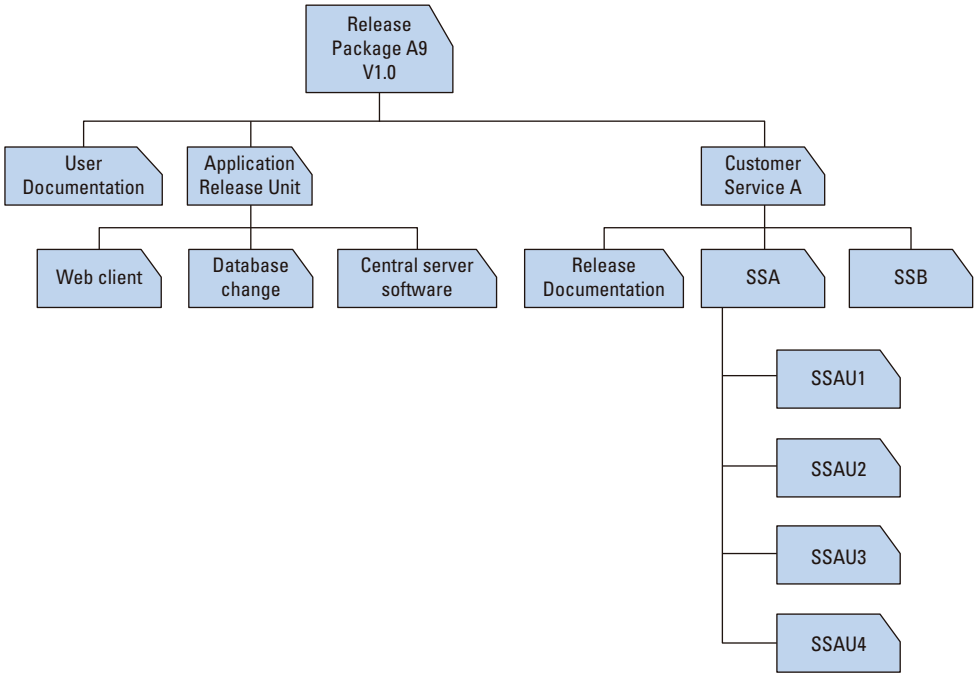

Figure 7.7 Sample of a release package

Source: The Cabinet Office

Release and deployment models

Release and deployment models define:

- Release structure
- The exit and entry criteria
- Controlled environments for builds and tests
- The roles and responsibilities for each CI
- The release promotion and configuration baseline model
- Template release and deployment schedules
- Supporting systems, tools and procedures for documenting activities

Activities, methods, and techniques

There are four phases to release and deployment management:

Table 7.9 The four phases to release and deployment management

Release and deployment planning	Plans for creating and deploying the release are created
Release build and test	The release package is built, tested and checked into the DML
Deployment	The release package in the DML is deployed to the live environment
Review and close	Experience and feedback are captured, performance targets and achievements are reviewed and lessons are learned

1. Release and deployment planning

Prior to a deployment into production different plans are formulated. The type and number depends on the size and complexity of the environment and the changed or new service. Release and deployment plans form part of the overall Service Transition plan. Change management will approve or reject the plans.

The plans will include the following:

- Scope and content of the release
- Risk assessment and risk profile for the release
- Organisations and stakeholders affected by the release
- Stakeholders that may authorise the change request for each stage of the release
- Team responsible for the release
- Deployment schedule for the release
  - Approach to working with stakeholders and deployment groups to determine:
    - Delivery and deployment strategy
    - Resources for the release build, test and deployment, and for early life support
    - Amount of change that can be absorbed.

The following (sub) plans are relevant for release and deployment:

- **Pass/fail criteria** – Service Transition is responsible for planning the pass/fail situations for every phase of the release and deployment.
- **Building and test plans** – Building and test plans are needed that describe the approach to the building, testing, and maintenance of the pre-production environment; different environments are needed for each test type: for unit testing, service release testing and integration testing.
  - Developing build plans from the SDP, design specifications and environment configuration requirements
  - Establishing the logistics, lead times and build times to set up the environments
  - Defining a configuration baseline for the build environment, to ensure that each build is carried out in a known environment
  - Testing the build and related procedures
  - Scheduling the build and test activities

- Assigning resources, roles and responsibilities to perform key activities
- Defining and agreeing the build exit and entry criteria

Controlled environments
Various controlled environments will need to be built or made available for the different types and levels of testing as well as to support other transition activities such as training

Planning release packaging and build
Planning the release packaging and build activities includes developing mechanisms, plans, or procedures

Preparations for release build and test
Before authorising the release build and test stage, the service design, and the release design must be validated against the requirement for the new or changed service offering. Record, track and measure any risks and issues against the services, service assets and CIs within the service package, SDP or release package.

Deployment planning
There are many planning considerations. Planners should be able to answer the following questions.
- What needs to be deployed?
- Who are the users?
- Are there location dependencies?
- Where are the users?
- Who else needs to be prepared well in advance?
- When does the deployment need to be completed?
- Why is the deployment happening?
- What are the critical success factors and exit criteria?
- What is the current capability of the service provider?

Logistics and delivery planning
Once the overall deployment approach is understood, develop the logistics and delivery plans.

Planning of pilots
A pilot can be used to establish the viability of most, if not all, aspects of the service.

Financial/commercial planning
Specifically checked before the deployment, and activities should be added to the deployment plans where necessary.

2. Release build and test
During the release build and test stage, the common services, and infrastructure need to be managed carefully, since they can significantly affect the build and test of a technology-enabled service and its underlying technology infrastructure.

Release and build documentation
Procedures, templates, and guidance should be used to enable the release team to take service assets and products from internal and external suppliers and build an integrated release package efficiently and effectively.

Acquire and test input configuration items and components
Configuration items and components are acquired from projects, suppliers, partners, and development groups. Historical information should be consulted when repetitive releases of same or similar CI are performed. There should be documented procedures and templates (models) for purchasing, distributing, installing, moving, and controlling assets and components.

Release packaging
Build management procedures, methodologies, tools and checklists should be applied to ensure that the release package is built in a standard, controlled, and reproducible way in line with the solution design defined in the service design package.

Build and manage the test environments
Effective build and test environment management is essential to ensure that the builds and tests are executed in a repeatable and manageable manner. Preparation of the test environments includes building, changing, or enhancing the test environments ready to receive the release.

Service testing and pilots
The testing activities are coordinated through test management. Testing aims to build confidence in the service capability prior to final acceptance during pilot or early life support. It will be based on the test strategy and model for the service being changed.

Service rehearsals
A service rehearsal (sometimes referred to as "model office" or a "table top") is a simulation of as much of the service as possible in an extensive and widely participatory practice session. A service rehearsal takes place just before deployment of the service but not too early or too close to the live date.

Pilots
A pilot sets out to detect if any elements of the service do not deliver as required and to identify gaps/issues in service management that put the service and/or the customer's business and assets at risk. As far as possible it should check that the utilities are fit for purpose and the warranties are fit for use.

3. Deployment

Plan and prepare for deployment
The planning and preparation activities prepare the group for deployment. This is an opportunity to prepare the organisation and people for organisational change.

During the actual deployment stage the detailed implementation plan is developed. This includes assigning individuals to specific activities.

Assess readiness of target group
The assessment of the deployment should be conducted early and periodically. The results of this assessment are provided as part of the detailed implementation plans for the target deployment group.

Develop plans
This activity includes assigning specific resources to perform deployment and early life support activities. Identify and assess risks specific to this deployment group by using the service model to identify business and service-critical assets that have the highest risk of causing disruption.

Perform transfer, deployment, and retirement
The following activities provide an example of the different aspects that will be performed in the order specified on the deployment plan.

Change/transfer financial assets
Changes and transfers of financial assets need to be completed as part of deployment.

Transfer/transition business and organisation
The transfer of a business unit, service or service unit will involve changing the organisation. When the change includes a transfer of service provider such as outsourcing, insourcing, or changing outsourcing providers, consider the following organisational elements: organisational change, quick wins, and communication.

Deploy processes and materials
Deploy or publish the processes and materials ready for people involved in the business and service organisation change. The materials may include policies, processes, procedures, manuals, overviews, training products, organisational change products, etc. Training people to use new processes and procedures can take time.

Deploy service management capability
This activity involves the deployment of the new or changed processes, systems, and tools to the service provider teams responsible for service management activities. Ensure personnel involved are competent and confident in operating, maintaining, and managing the service in accordance with the service model.

Transfer service
Transferring a service will also involve organisational change described earlier in this section.

Deploy service
This activity performs the deployment of the service release and carries out the activities to distribute and install the service, supporting services, applications, data, information, infrastructure, and facilities.

Decommissioning and service retirement
Some specific aspects need to be considered for decommissioning and retiring services and service assets. Procedures for retiring, transferring, or redeploying service assets should consider security, licensing, environmental, or other contractual requirements.

Remove redundant assets
Identify and remove redundant assets, thereby potentially saving licence fees, liberating capacity, and preventing accidental use.

Verify deployment
It is important to verify that users, service operation functions, other personnel, and stakeholders are capable of using or operating the service.

Remediate/back out release
The most common form of remediation is to back out the release, restoring all hardware, software, and data to the previous baseline. Alternative forms of remediation include implementing normal changes or emergency changes to resolve problems, or invoking IT service continuity plans to provide the service.

Early life support
Early life support (ELS) provides the opportunity to transition the new or changed service to service operation in a controlled manner and establish the new service capability and resources. Formal handover of the new or changed service to the service operation functions happens in two stages. At the beginning of early life support there should be a formal notification that the service is now in live use. At the end of early life support there should be a formal notification that all SLAs are now being enforced and the service is fully operational.

4. Review and close
In the review of a deployment, check whether:
- The knowledge transfer and training were adequate
- All user experiences have been documented
- All fixes and changes are complete and all problems, known errors and workarounds have been documented
- The quality criteria have been complied with
- The service is ready for transition from ELS into production

Also check whether there are issues that have to be passed on to CSI. The deployment is completed when the support is transferred to Operations. Finally, change management will conduct a Post Implementation Review (PIR).

To finalise the Service Transition as a whole a formal change evaluation must be performed that is tailored to with the scale and scope of the change.

Information management

Throughout the release and deployment management process, appropriate records will be created and maintained. As configuration items are successfully deployed, the CMS will be updated with information such as:

- New, changed, or removed configuration items
- Relationships between requirements and test cases
- New, changed, or removed locations and users
- Status updates
- Change in ownership of assets
- Licence holding

Other data and information will also be captured and recorded within the broader service knowledge management system. This could include:

- Release packages in the DML
- Installation/build plans
- Logistics and delivery plans
- Validation and test plans, evidence and reports
- Deployment information, deployment history, who was involved, timings, etc.
- Training records
- Access rules and levels
- Known errors

As part of the deployment clean-up activities it is important to delete or archive redundant records related to the previous service or products.

Interfaces

Like all processes, the release and deployment management process has relationships with all other processes. However, some of the most visible relationships include but are not limited to the following.

Design coordination

- The design coordination process creates the service design package that defines the new service, including all aspects of how it should be created
- Plans and packages should be developed and documented during the service design stage, and design coordination will ensure that these are documented in the SDP

Transition planning and support

- Provides the framework for release and deployment management to operate in, and transition plans provide the context for release and deployment plans

Change management

- Provides the authorisation for the work
- Release and deployment management provides the execution of many changes

- Release and deployment plans are a significant part of the change schedule
- Deployment review is often combined with the review and closure of the change

Service asset and configuration management

- Release and deployment management depends on data and information in the CMS, and provides many updates to the CMS. It is important that these updates are coordinated and managed properly as otherwise the data will not be kept up to date.

Service validation and testing

- To ensure that testing is carried out when necessary
- To ensure that builds are available when required by service validation and testing

Triggers

Here are some of the triggers that may start the whole, or part of the, release and deployment management process.

- Starts with receipt of an authorised change to plan, build, and test a production-ready release package. Deployment starts with receipt of an authorised change to deploy a release package to a target deployment group or environment

Inputs

Here are some of the inputs required by the release and deployment management process.

- Authorised change
- Service design package
- IT service continuity plan and related business continuity plan
- Service management and operations plans and standards
- Technology and procurement standards and catalogues
- Acquired service assets and components and their documentation
- Build models and plans
- Environment requirements and specifications for build, test, release, training, disaster recovery, pilot and deployment
- Release policy and release design from service design
- Release and deployment models including template plans
- Exit and entry criteria for each stage of release and deployment management

Outputs

Here are some of the outputs produced by the release and deployment management process.

- New, changed or retired services
- Release and deployment plan
- Updates to change management for the release and deployment activities
- Service notification
- Notification to service catalogue management to update the service catalogue with the relevant information about the new or changed service
- New tested service capability and environments

- New or changed service management documentation
- SLA, underpinning OLA, and contracts
- New or changed service reports
- Tested continuity plans
- Complete and accurate configuration item list with an audit trail for the CI in the release package and also the new or changed service and infrastructure configurations
- Updated service capacity plan aligned to the relevant business plans
- Base-lined release package – checked in to DML and ready for future deployments
- Service transition report

Critical success factors

The critical success factors, which change over time, for the release and deployment management process may include but are not limited to the following.

- Defining and agreeing release plans with customers and stakeholders
- Ensuring integrity of a release package and its constituent components throughout the transition activities
- Ensuring that the new or changed service is capable of delivering the agreed utility and warranty
- Ensuring that there is appropriate knowledge transfer

Metrics

The performance of the release and deployment management process can be measured according to:

- Increased…
  - Number and percentage of releases that make use of a common framework of standards, re-usable processes and supporting documentation
  - Number and percentage of releases that meet customer expectations for cost, time and quality
  - Score in surveys of customer, user, and service operation function satisfaction with release and deployment management
  - Customer and user satisfaction with the services delivered
- Reduced…
  - Number of CMS and DML audit failures related to releases
  - Number of deployments from sources other than the DML
  - Number of incidents due to incorrect components being deployed
  - Variance from service performance required by customers
  - Number of incidents against the service
  - Customer dissatisfaction
  - Resources and costs to diagnose and fix incidents and problems in deployment and live use
  - Number of incidents categorised as “user knowledge”

Challenges

Here are some of the potential challenges faced by the release and deployment management process

- Developing standard performance measures and measurement methods across projects and suppliers
- Dealing with projects and suppliers where estimated delivery dates are inaccurate and there are delays in scheduling service transition activities
- Understanding the different stakeholder perspectives that underpin effective risk management for the change impact assessment and test activities
- Building a thorough understanding of risks that have impacted or may impact successful service transition of services and releases
- Encouraging a risk management culture where people share information and take a pragmatic and measured approach to risk

Risks

Here are some of the potential risks faced by the release and deployment management process.

- Poorly defined scope and understanding of dependencies in earlier lifecycle stages
- Using personnel who are not dedicated to release and deployment management activities
- Failing to use the release and deployment management process to manage service retirement
- Lack of…
  - Integration with the appropriate financial cycles and activities
  - Integration with the appropriate corporate governance, regulatory controls, and requirements regarding licensing and security
  - Operational support
  - Consideration for all capabilities and resources
  - Consideration for people, process, products and partner aspects
- Not managing or addressing organisational and stakeholder change:
- Poor commitment and decision-making
- Failure to obtain appropriate authorisation at the right time
- Indecision or late decision-making
- Inadequate or inaccurate information
- Health and safety compromised
- Time allowed for release and deployment management
- Not managing suppliers/sourcing/partnering relationships during transition:
- Inadequate “back-out” or “contingency” plan if sourcing/partnering fails

7.9 Service Validation and Testing

Introduction

Testing of services is an important contribution to the quality of IT service provision. Testing ensures that new (or changed) services are **fit for** purpose and **fit for use**.