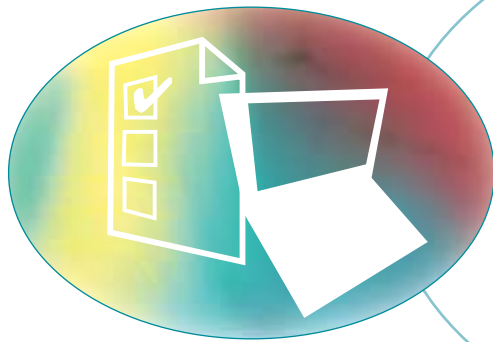




Release Management

Release Management Contents

RM 1	Topic introduction – Aim and objectives of this topic	1
RM 2	Overview – An introduction to the process	1
RM 3	Implementation guide – How to implement the process	5
RM 4	Operations guide – The ongoing operation of the process	14
RM 5	Roles and responsibilities	17
RM 6	Review – Summary and checklist	18
Appendices		22
Glossary		31

Key

Glossary term: [Glossary term](#)

Cross reference: [Cross reference](#)



Release Management

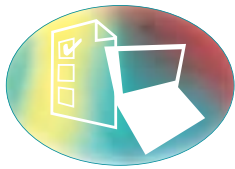
© Becta 2004

You may reproduce this material free of charge in any format or medium without specific permission, provided you are not reproducing it for profit, material or financial gain. You must reproduce the material accurately and not use it in a misleading context. If you are republishing the material or issuing it to others, you must acknowledge its source, copyright status and date of publication.

Publication date March 2004

Originally published online in September 2003 as part of the Becta website
<http://www.becta.org.uk/tsas>

While every care has been taken in the compilation of this information to ensure that it is accurate at the time of publication, Becta cannot be held responsible for any loss, damage or inconvenience caused as a result of any error or inaccuracy within these pages. Although all references to external sources (including any sites linked to the Becta site) are checked both at the time of compilation and on a regular basis, Becta does not accept any responsibility for or otherwise endorse any product or information contained in these pages, including any sources.



Release Management

RM 1 Introduction to Release Management

Do you want to know how to roll out new software and hardware efficiently and effectively? FITS Release Management tells you how.

- RM 1.1 Aim
- The aim of this section is to introduce Release Management and to help you implement the process in your school with a minimum of preparation and training.
- RM 1.2 Objectives
- The objectives of this section are to enable you to:
- understand the concept and benefits of Release Management
 - understand what is involved in the process of Release Management
 - understand the roles and responsibilities in Release Management
 - implement a basic Release Management process in your school
 - continue to operate this Release Management process
 - identify useful measurements to gain benefit from the Release Management process you have implemented
 - review your implementation and summarise your progress.

RM 2 Overview

RM 2.1 What is Release Management?

What is Release Management?

Release Management is the process of planning, building, testing and deploying hardware and software and the version control and storage of software.

Its purpose is to ensure that a consistent method of deployment is followed. It reduces the likelihood of incidents as a result of rollouts and ensures that only tested and accepted versions of hardware and software are installed at any time.

Why use Release Management?

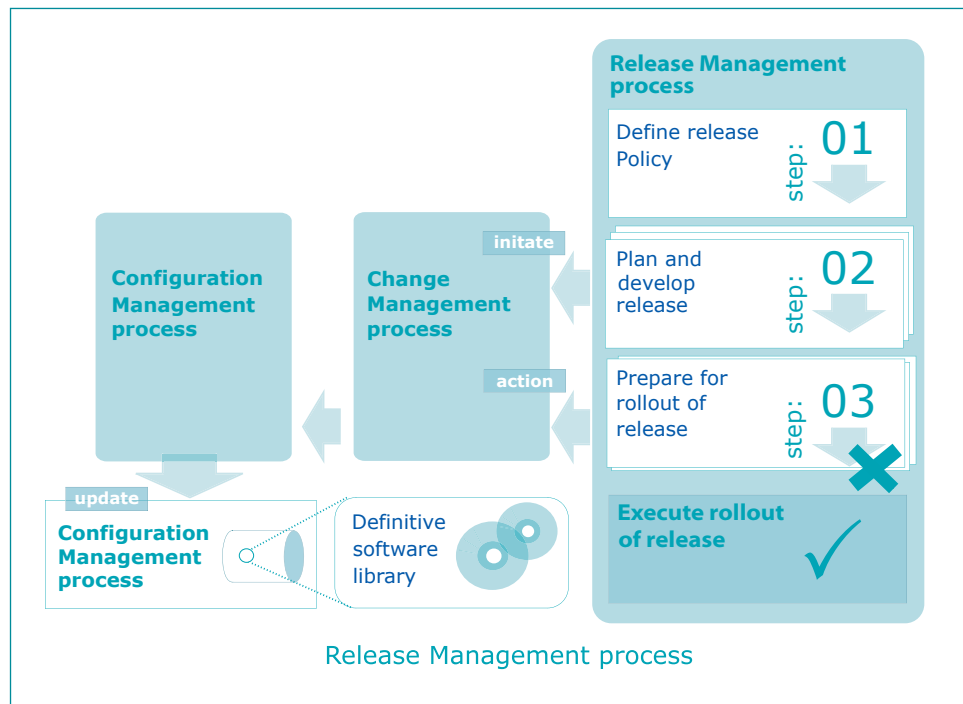
Release Management is proactive technical support focused on the planning and preparation of new services. Some of the benefits are:

- the opportunity to plan expenditure and resource requirements in advance
- a structured approach to rolling out all new software or hardware, which is efficient and effective
- changes to software are 'bundled' together for one release, which minimises the impact of changes on users
- testing before rollout, which minimises incidents affecting users and requires less reactive support

- an opportunity for users to accept functionality of software before it is fully implemented
- training in advance of rollout, which means that users do not experience system downtime while learning new features
- version control and central storage of software, ensuring that correct versions are installed at all times, which minimises incidents and the need for reinstallation.

Who uses Release Management?

Those responsible for ICT and ICT technical support use Release Management. These may include external suppliers.



RM 2.2

How Release Management works

The Release Management process works by providing a consistent framework for defining and creating new services, and ensuring that the correct versions of tested and approved software are implemented on a day-to-day basis (that is, after initial rollout).

It interfaces with the **Change Management** process to enable implementation and to the **Configuration Management** process to maintain configuration records.

The Release Management process flowchart (above) illustrates this.

For further details see the sections below:

RM 2.2.1

Release policy

Release policy is the strategy adopted for implementing new services. It can be complex or simple.

At a simple level, release policy may be the conscious decision to implement new computers only twice a year, for example, or to upgrade software in phases in a certain order such as by department. It is a high level plan that is agreed and published in advance to set expectation.

At a complex level, release policy can relate to the actual development of software and determine frequency of new versions, version-numbering convention, types of

release such as full or partial, and so on. This applies more specifically to organisations with their own software development function.

RM

2.2.2

Definitive software library

The **definitive software library (DSL)** is a repository for storing released software and serves as the central point for obtaining versions of software for installation. Its purpose is to distinguish between old and new released versions and any development software.

The definitive software library links to the **configuration management database**.

RM

2.2.3

Release planning

Release planning is proactive technical support to ensure that software or hardware being deployed to users does what it is required to do when they receive it. Release planning includes design, build, test and acceptance.

Design	Design relates to the architecture of a new ICT service. That is the configuration of the pieces of hardware and software involved.
Build	Build is the compilation of components to form the service, such as the installation and set up of a new computer or the integration of a new piece of software with existing applications on the desktop.
Test	Test is the internal technical support testing that should be carried out to ensure that the service is stable and that other services have not been affected by its introduction.
Acceptance testing	Acceptance testing is functionality testing carried out by nominated users who are responsible for ensuring that the service does what is required.

RM

2.2.4

Release rollout

Release rollout is the actual deployment of the new hardware or software. The term 'rollout' implies the introduction of a new service to all or many computers or users, but release management techniques can and should be applied to individual installations. Release rollout includes scheduling, training, communication, updating the **definitive software library** and checklists.

Scheduling	Scheduling is required for both the actual deployment of the service and any training that is required. Scheduling is particularly important for rollouts to more than one computer or user, but even one installation should be scheduled in accordance with the user's preference.
Training	Training should always be considered prior to a rollout of new hardware or software. It may not always be necessary – for example if the rollout is of new computers to existing computer users – but it is good practice to take the potential need for training into account.
Communication	Communication is very important. Users need to be made aware that changes are being planned and how the schedule affects them.

Update definitive software library	The correct version of software to be installed must be available and marked clearly in the definitive software library in time for rollout.
Checklists	The actual deployment should also be structured to ensure that all activities are completed and the rollout is consistent. Checklists should be available for the installers to follow.

RM

2.2.5

Relationships with other processes

Relationship with Change Management	The Release Management process interfaces with the Change Management process throughout its lifecycle. Release management provides the inputs to the request for change at the various stages of planning and preparation. Change management final approval should be received before the new service is deployed into the live environment.
Relationship with Configuration Management	The Release Management process links closely to Configuration Management . The final step in the release of a new service or an upgrade to an existing service is to record the changes in the configuration management database . This is facilitated by the Change Management process or the incident/request process as appropriate. The definitive software library is also considered to be part of the configuration management database.

RM

2.3

What does Release Management cost?

Release Management can cost as little or as much as you can afford. There are three aspects to consider: expenditure, people and time.

You incur financial expenditure only if software tools are purchased for the **definitive software library** and additional hardware is required to accommodate it. Specific products are available to manage security and version control, but these are only really needed in a software development environment or regulated industry such as pharmaceuticals.

Release Management requires full time staff only if there is a large volume of new services and upgrades requiring full-time effort for planning and execution. This sometimes applies in large organisations. In a school, Release Management activities should be part time and you should be able to allocate the roles and responsibilities to existing members of staff. Roles and responsibilities are referred to throughout the Release Management section, and are grouped together in **RM 5 Roles and responsibilities**.

The amount of time taken up by the Release Management process once it is operational is difficult to quantify, as this will depend on the frequency of release of new services in your school. It is worth investing time in the planning stages of introducing new services and implementing them in a consistent way to avoid the disruption and downtime that may occur later if they are badly executed. It is always better to spend time on prevention rather than cure.

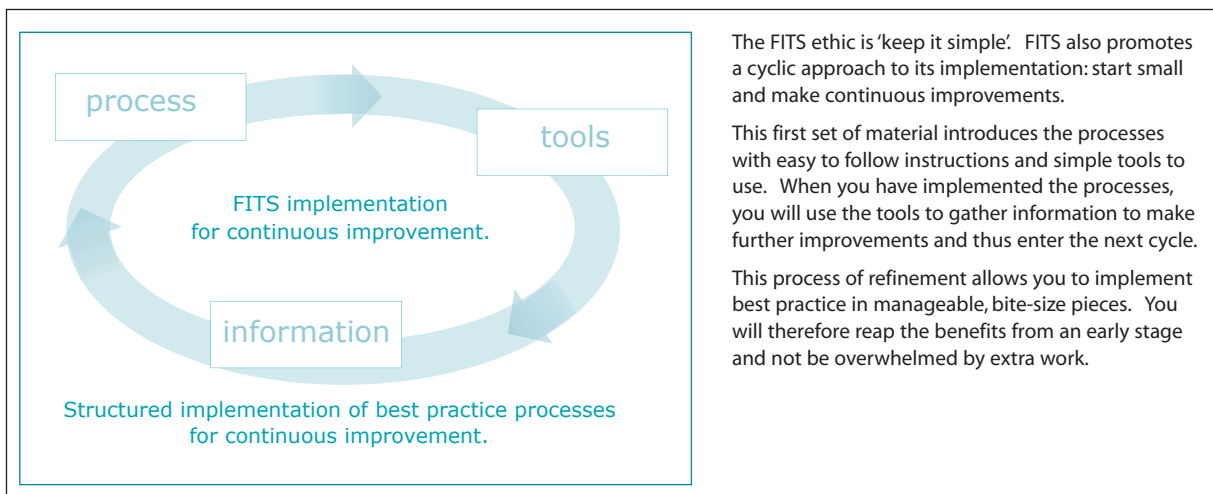
Remember to allow time also for the implementation and integration of the process into normal day-to-day activities. We have created a table of activities to help you plan the amount of time required.

Activity	Example	Further information
Preparing for implementation	Discussions, planning	RM 3 Implementation guide
Implementation	Training, pilot, actual implementation	RM 3 Implementation guide
Review of implementation	Difficulties with process or roles	RM 3 Implementation guide
Preparing releases	Designing, building, testing and acceptance testing of releases of hardware or software	RM 4 Operations guide
Carrying out installations	Scheduling and executing training, scheduling deployment, communicating with users, preparing checklists for installers, deployment of releases.	RM 4 Operations guide
Monitoring the process	Reporting against the process and ensuring that it is effective	RM 4 Operations guide

RM 3 Implementation guide

Define what needs to be done

As described in the overall FITS implementation approach, we recommend a phased approach to implementing new processes.



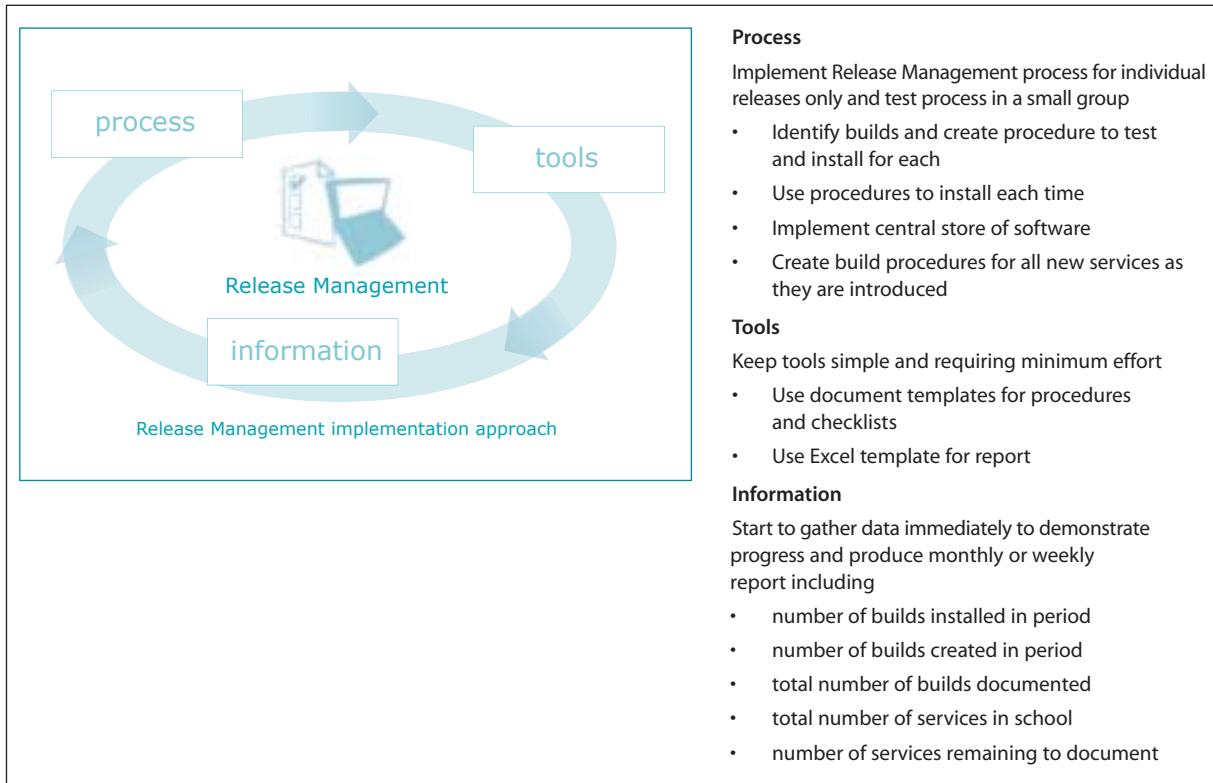
The FITS ethic is 'keep it simple'. FITS also promotes a cyclic approach to its implementation: start small and make continuous improvements.

This first set of material introduces the processes with easy to follow instructions and simple tools to use. When you have implemented the processes, you will use the tools to gather information to make further improvements and thus enter the next cycle.

This process of refinement allows you to implement best practice in manageable, bite-size pieces. You will therefore reap the benefits from an early stage and not be overwhelmed by extra work.

FITS Release Management is for people with little free time to spend on implementing processes and procedures and whose day-to-day activities are unpredictable and must take priority.

Our aim is to help you begin to remove some of the unpredictability by introducing best practice processes in small steps and so begin to realise the benefits as quickly as possible.



Long-term scope

In the long-term, Release Management should be applied as a strategy for introducing all new software or hardware in a planned, controlled and structured manner. This should therefore reduce the need for ad hoc requirements as far as possible and allow technical support time to be focused on other activities. It also results in economies of scale as the planning and preparation activities do not increase in proportion to the number of items – these tasks must be performed whether the exercise is to install one computer or 10.

However, to ensure the best chance of success, application of Release Management to this extent is best left until the basic concepts are fully understood and automatic.

Short-term scope

In the short-term, Release Management should be applied to the installation of single instances of hardware or software. This exercise can be used to begin the generation of standard builds and a centralised store of software and introduces the concept of a standard process for implementing all equipment. In addition, by constraining the Release Management process to single installations, you can avoid the need to interface to the [Change Management](#) process. This means that Release Management can be implemented in isolation with no prerequisite for a Change Management process.

Once the basic activities are comfortably in place, it will be a relatively easy step to consider the bigger picture and to plan and manage larger rollouts that will require input to the Change Management process.

Prepare to implement

Good preparation can make the difference between a successful implementation of a process and an unsuccessful one.

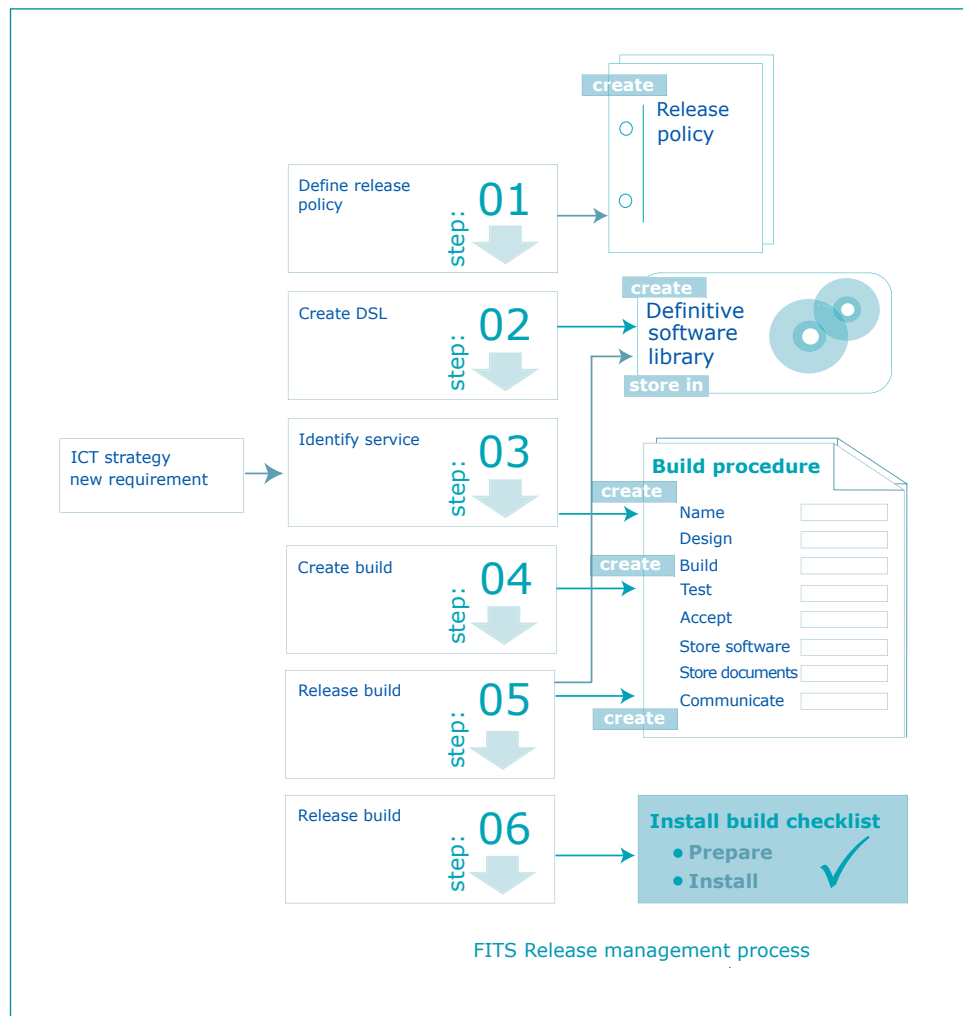
Roles and responsibilities	The first step is to identify the process participants and assign roles and responsibilities. We recommend that for initial implementation you involve as few people as possible so that it can become familiar with minimum impact on the day-to-day workload of the school. Who you select to fulfil the Release Management roles will depend on how you currently provide technical support and who is involved already. RM 3.2.1 Assigning roles and responsibilities in Release Management offers some suggestions and guidance.
Training	After you have assigned roles and responsibilities, it is important to ensure that those participating in the implementation and subsequent operation of the process understand what is required of them. Use the FITS website as training material.
Start date	Set a start date. A 'go-live' date is important in any implementation. Make sure that you allow enough time for all the other preparatory tasks to be carried out before your 'go-live' date.
Communication	Communication must take place within the implementation team to agree plans, schedule dates, and so on, but it is also important to communicate externally and inform the user community of the new process and its benefits to them. The implementation of a process can be seen as being a change just like the upgrading of a server and the impact on the user community should be communicated to them clearly in advance of the change. They will more readily embrace it if they are not taken by surprise.
Materials	Before you can go ahead with the implementation, you will need all the materials required for the process. Make sure that you have downloaded the templates you need and that everyone involved has access to them.
Pilot	Carry out a pilot implementation as a test first. Practise creating a build procedure by installing a piece of software on a technical support computer. Then use the procedure and standard installation checklist to install it again. Follow all the instructions right down to storing the software in the definitive software library. It is important to test every aspect of the process in the pilot so that you can be sure that there are no outstanding issues.
Prerequisites	FITS Release Management has been designed to be implemented on its own with no prerequisites. However, the processes Release Management most closely interfaces with are Change Management and Configuration Management. Change Management's role in Release Management is in the approval and rollout of large releases to more than one end-user and, as such, is not recommended at this stage of implementing Release Management. Configuration Management is not essential at this stage either, but may be helpful in identifying the different services that need to be documented in Release Management. There are no strict prerequisites therefore and, if you wish, you can implement FITS Release Management in isolation.

Role	Suggested representative(s)	Comments
Release manager	Person with overall responsibility for Release Management or ICT in general, eg: • ICT manager • ICT co-ordinator • network manager • technician.	This may be delegated to someone in the ICT team but there should be only one release manager for the sake of accountability.
Build developer	Person responsible for integrating new hardware or software into existing services, eg: • technician • ICT co-ordinator • network manager • supplier.	Build developers should be technical enough to install and test new hardware and software and also resolve any conflicts with existing services that may arise. Build developers and installers may be the same person.
Acceptance tester	Person responsible for confirming that the new hardware or software performs the functions it was obtained for, eg: • teacher • teaching assistant • administrative staff • any end-user.	Acceptance testers should vary depending upon the service being tested. Someone with intimate knowledge of the processes the hardware or software supports should perform this role. The acceptance tester shouldn't be the build developer or installer unless of course they are the end-user of the service or the service being tested is shared infrastructure.
Installer	Person responsible for performing day-to-day installations of hardware or software, eg: • technician • ICT co-ordinator • network manager • supplier.	An installer is likely to be the same person or people who carry out day-to-day technical support. You may have as many or as few installers as appropriate for your school. Build developers and installers may be the same person.

Implement

This section describes how to implement a basic release management process and the tools required to support it.

- Step 1: Define policy
- Step 2: Create definitive software library
- Step 3: Identify service
- Step 4: Create build
- Step 5: Release build
- Step 6: Install build



Step 1 Define policy

The release policy is something that should be determined locally. However, in order to implement Release Management effectively, we recommend the following initial policy.

- Create and follow a standard process for installing and testing new user hardware and software.
- Create and follow a standard procedure for each different hardware and software service.
- Store all software centrally.

Step 2 Create definitive software library

In basic terms, a **definitive software library** is the central storage and control of original software, build software and software licences – for more details of which see overleaf.

You should make it a policy to create and develop a definitive software library to help you manage your software licence allocation and ensure that the correct versions of software are installed. It also saves time searching for original disks when you are in a hurry to install something. It is about tidiness as much as anything else.

Original software

Always retain original software – at least one set of each version – in a central library.

Storage	Diskette or CD boxes provide adequate storage for original software
Location	Keep all the boxes together in a lockable cupboard.
Security	Ensure that only those authorised to install software have access. Track comings and goings with an in/out book and ensure that all staff record when and what they have taken and when it has been returned.

Build software

Build software can be defined as working copies of software, either stored on a file server, on a boot disk/CD or as a computer disk image. If you keep build software, you must ensure that such use complies with your software licence agreements.

Storage	If you use a file-server for storage, create a common area for all such copies with a logical folder structure. See Appendix A for our example DSL folder structure. Disks or CDs should be kept as described in the section below on original software.
Location	For storage purposes, you can use any file server that can be accessed from all points for installation purposes. It is often the case that build software is installed on a file server used only by ICT staff. This is so that, if necessary, it can be taken offline without impact on normal user activities. Disks or CDs should be kept as described in the section above on original software.
Security	File-server areas should have access restricted to those responsible for release software builds and those authorised to install them. Full access should be granted only to those responsible for release. Installers should have read-only access rights. It is important to put in place restrictions to demonstrate that attempts have been made to control access and preserve integrity. This is where software tools designed specifically to manage software releases demonstrate their benefits. Disks or CDs should have the same security applied as described in the section above on original software.

Software licences

Compliance with software licensing rules is very important. It can be difficult sometimes to keep track of how many the school owns and how many are in use.

A simple way to do this is to create a spreadsheet or document with an entry for each licence and then record the assignment of a licence each time it is installed. We have created a licence list template (see [Appendix B](#)) for you to download and use. Keep a separate list for each operating system or application and don't forget to add new licences to the list when you buy them.

Storage	Store the physical licences somewhere safe and keep them all together. For example, you could store them in a locked cupboard with the original software (see above). Keep the licence list in the definitive software library. See our example DSL folder structure (Appendix A) for ideas.
Location	The definitive software library and licence lists should be on a file server that can be accessed by those responsible for assigning licences and installing software.
Security	The definitive software library and licence lists should be accessible only by those authorised. It is important to put in place restrictions to demonstrate that attempts have been made to control access and preserve integrity.

Step 3 Identify service

A service is the specific hardware or software type to be implemented – for example:

- a desktop computer
- a laptop computer
- a software application
- an operating system.

Each different service will have a separate release management procedure.

Select a suitable service with which to implement Release Management and your first procedure. This should be a service that you are required to install currently and you should follow the procedure through to installation. This will ensure that you have tested all the steps before handing over the procedure to incident management staff. You will gain the most benefit from starting with a frequently requested service.

It may also help you to gain an approximate view of the overall number of services currently in use. This in turn will help you to decide which ones to start with and will give you a [baseline](#) from which to measure progress. See [RM 4.2.3 Monitoring the Release Management process](#) for more information on measurements.

Step 4 Create build

A 'build' is the service to be installed actually in working order. For example, the build of a piece of software is its actual installation on the computer and its interaction with the hardware and other software to deliver the service required. A hardware build might be the connecting together of the base unit, monitor, keyboard and mouse, the installation of the operating system and standard software and the setting up of printer drivers to enable printing to a shared printer. In other words, a 'build' is a working service as opposed to its component parts still in boxes.

By going through the create build steps (see overleaf) for a specific service, you will identify the procedure to follow each time the service has to be installed.

To enable the process, we have created a build procedure template (see [Appendix C](#) for hardware or [Appendix D](#) for software) for you to use as the starting point and add to with the specific details of each service. Save the template as a separate document for each build and enter the name of the document in the footer. We have also created an example build procedure for hardware (see [Appendix C](#)) and an example build procedure for software (see [Appendix D](#)) to help you to understand the requirements of the document and demonstrate how it can be used for either type of service.

Create build steps

Using the build procedure template (see [Appendix C](#) for hardware or [Appendix D](#) for software), complete the sections as described below. See also the example build procedure for hardware (see [Appendix C](#)) and example build procedure for software (see [Appendix D](#)).

Section 1: Design	List the components of the service and include any relevant comments.
Section 2: Build	Describe as fully as possible the steps required to install the service.
Section 3: Test	List all other services that must remain stable in the same environment and confirm successful test. Note that a build cannot be released until all services are stable.
Section 4: Accept	Identify, in conjunction with an appropriate user or group of users, suitable criteria for functionality acceptance. Describe the acceptance tests as fully as possible.

Step 5 Release build

A 'build' is the service to be installed actually in working order. For example, the build of a piece of software is its actual installation on the computer and its interaction with the hardware and other software to deliver the service required. A hardware build might be the connecting together of the base unit, monitor, keyboard and mouse, the installation of the operating system and standard software and the setting up of printer drivers to enable printing to a shared printer. In other words, a 'build' is a working service as opposed to its component parts still in boxes.

The first step in the Release Management process is to create the build for the particular service – see Step 4. When the build has been tested and accepted, it can be released – that is, made available for installation using the build procedure for that service (see section below on release build steps).

Release build steps

Using the build procedure template (see [Appendix C](#) for hardware or [Appendix D](#) for software), follow the instructions for the sections as described below. See also example build procedure for hardware ([Appendix C](#)) and example build procedure for software ([Appendix D](#)).

Section 5: Store software in DSL	In the definitive software library (DSL) list the names and versions of all software included in this build and note the path to the location of the master copy. Physically store the software in the location described and, when you've done, tick in the box provided on the build procedure.
Section 6: Store documents and record in CMDB	Record the name and version of the build procedure. The build procedure should be stored centrally – record the path and save the document to that location under the chosen file name. Don't forget to tick the box. Also store all user documentation centrally and record the location. If/when you have a configuration management database, you should enter details there too.
Section 7: Communicate availability	List all the roles authorised to install hardware or software and confirm that they have been informed of the new build. See example build notification (Appendix E) for a suggested form of words.

Step 6 Install build

A 'build' is the service to be installed actually in working order. For example, the build of a piece of software is its actual installation on the computer and its interaction with the hardware and other software to deliver the service required. A hardware build might be the connecting together of the base unit, monitor, keyboard and mouse, the installation of the operating system and standard software and the setting up of printer drivers to enable printing to a shared printer. In other words a 'build' is a working service as opposed to its component parts still in boxes.

Before a build can be installed, you must create build ([Step 4](#)) and release build ([Step 5](#)). If the build has already been created and released, you can install it in accordance with the install build steps (see below).

Because these steps are fundamentally the same for all installations, we have created an install build checklist template (see [Appendix F](#)) to help you carry out this stage.

Install build steps

Use the following guidelines to complete the install build checklist template (see [Appendix F](#)). See also our example install build checklist ([Appendix F](#)).

Section 1: Install details	Complete the checklist with the details specified. Liaise with the user to schedule appropriate times for installation and training.
Section 2: Install checklist	Use the checklist to ensure that all steps are followed and tick the boxes to confirm that this is the case.
Section 3: Install document details	Save the install build checklist as a separate document for each install and enter the name and location of the file here.

RM

3.4

Review the implementation

After you have created your first build procedure and followed the procedure through to installation, stop before you do any more. Ask some key questions and consider the answers before continuing to use the process.

- Did everyone understand what was required of them?
- Was the build procedure template easy to follow and complete?
- Does training need to be revisited before continuing?
- Were those responsible for installing hardware and software aware of the existence of the build procedure?
- Does everyone who should know how to access software?
- Was the licence list updated?
- Was everyone informed of the new process?

RM

3.5

Implementation resources

- For creating a build procedure, use the build procedure template (see [Appendix C](#) for hardware or [Appendix D](#) for software)
- For installing a build, use the install build checklist template (see [Appendix F](#))
- For creating a licence list, use the licence list template (see [Appendix B](#))

RM 4 Operations guide

RM 4.1 What needs to be done?

The day-to-day operational tasks in Release Management are:

RM 4.1.1 Installing builds

Build installation requests should eventually, through the implementation of FITS, be generated in two ways:

Process	Type of request	Further details
Incident/request process	Individual user requests	Incident Management
Request for change process	Infrastructure requirements	Change Management

Before a build can be installed, you must create build (RM 3.3 Step 4) and release build (RM 3.3 Step 5). If the build has already been created and released, you can install it in accordance with the install build steps (see RM 3.3 Step 6)

RM 4.1.2 Creating new builds

New builds must be created as hardware and software is identified that has not yet had a build procedure created for it or if new types of hardware or software are introduced.

In either case the create build steps (see RM 3.3 Step 5) should be followed first and then the release build steps (see RM 3.3 Step 5).

RM 4.1.3 Monitoring the Release Management process

It is important to monitor the Release Management process to ensure that it is working. For example, the release manager needs to know that progress is being made on the number of standard build procedures being created and it is a good idea to get an overall view of the volume of installs being carried out on a regular basis, such as monthly.

The following measurements should be easy to gather and report on:

- number of builds installed in period
- number of builds created in period
- total number of builds documented
- total number of services in school
- number of services remaining to document.

To get started on some simple measurements and reporting, download our Release Management report template (see Appendix G) with graphs produced in Excel. Follow the instructions to fill in the volumes and check that the print range is set before printing.

See also Appendix G for our example Release Management report completed with dummy figures.

Making decisions

The purpose of the reports is to help you to make decisions. They must be interpreted to identify any areas for concern that need to be addressed. Remember that they are just statistics and should not be taken at face value. See them as the basis for asking questions, not as outright answers. Also, don't look at them in isolation: consider the bigger picture when reviewing reports and look at the reports from other processes, such as [Incident Management](#).

- A reduction in the number of build procedures created from one month to the next may be a result of a higher-than-usual volume of [incidents](#) keeping ICT staff busy, or it may be due to ICT staff absence, or it may be because there are no builds left to document.
- A reduction in the number of builds installed from one month to the next may be the result of a reduced number of requests or because there is a backlog of more urgent work.
- An indication that there are no services left to document may be true or it may mean that the total number hasn't been updated with new services introduced since the beginning of the Release Management implementation.

These are just examples to illustrate that statistics should not be taken at face value. Talk to the process participants and consider other related factors such as incident activity to understand the reality of the situation.

When does it need doing?

Installing builds

You need to install builds when:

- a user requests a new item of hardware or software
- a new infrastructure item is to be introduced.

Scenario 1

An individual user requests a single piece of software for use on their computer. A technician follows the build procedure and build install checklist for the application in question and installs it as requested.

Scenario 2

A new file [server](#) is required to increase capacity. A technician follows the build procedure and build install checklist to install the file server operating system.

Build procedures should be used always to ensure consistent installations and to help eliminate recurring incidents.

Creating new builds

You need to create new builds when:

- you receive a request for an installation that has not yet been through the process of testing and documenting.
- a request is received for a new type of hardware or software not previously installed.

Scenario 1

A new model of desktop computer is launched and models previously purchased by the school become obsolete.

When the first new one arrives, a new build procedure must be created to test the computer with software and other hardware already in use and to establish a standard procedure for installing subsequent deliveries.

Scenario 2

The ICT co-ordinator decides to implement a new Spanish language software product in the school's language laboratory.

The technician(s) involved in the project create a new build procedure for the software to ensure that it works as expected and doesn't cause problems with other applications.

The new procedure is later used in conjunction with the install build checklist to roll out the product to all of the language lab computers.

Ad hoc effort should be minimised where possible, by planning to create build procedures for all existing services.

RM

4.2.3

Monitoring the Release Management process

The Release Management process should be monitored regularly as soon as possible after introducing it. The initial period will focus on developing and documenting builds and it is important to see how much progress is being made.

Because developing and documenting builds is a proactive activity, time must be set aside to achieve it, although it is easy to let these types of activity go undone in favour of reactive work. It is the release manager's responsibility to ensure that sufficient time is devoted to the Release Management process.

We recommend that, once the release management activities have begun, our report be completed every week or month. The right frequency for your school will depend on the volume and frequency of your release management work. You may decide to start slowly and produce reports monthly but then ramp up to a higher turnover, at which point weekly reports may be more appropriate.

Match your reporting frequency to your work throughput and adjust the frequency in a planned way. Don't chop and change, as this will distort the measurements and make it difficult to see trends.

RM

4.2.4

Making decisions

All reports should be reviewed as soon as they are produced. It is important to identify issues as soon as possible so that corrective action can be taken.

If investigative work is required to identify the cause of issues, this will be harder if the trail is cold.

RM

4.3

Who does it?

Installers are responsible for installing builds. Build developers are responsible for creating new builds.

Monitoring the Release Management process is the responsibility of the release manager. Although the task of producing reports may be delegated to an administrator or technician, the release manager should retain the task of decision making in order to improve the process and the service it facilitates.

See [RM 3.2.1 Assigning roles and responsibilities in Release Management](#) for further information.

- RM 4.4 Operational resources
- Build procedure template (see [Appendix C](#) for hardware or [Appendix D](#) for software)
 - Install build checklist template (see [Appendix F](#))
 - Licence list template (see [Appendix B](#))

RM 5 Roles and responsibilities

- RM 5.1 Release manager
- Is responsible for Release Management and ensuring that the process is followed
 - Is the process owner
 - Should have some understanding of the hardware and software deployed
 - Does not need to be very technical
 - May be the person responsible for ICT and/or technical support
 - Some or all activities may be delegated to a technician
- RM 5.2 Build developer
- Creates new builds
 - Tests the stability of new builds and resolves any issues
 - Tests the impact of new services on existing components of the build and resolves any issues
 - Creates build procedures for all new hardware and software installation
 - Stores software in the [definitive software library](#)
 - Stores technical and user documentation appropriately
 - Liaises with acceptance testers (see [RM 5.3](#)) to ensure service functionality
 - Ensures that installers (see [RM 5.4](#)) are aware of new build procedures
 - Supports installers in the event of difficulties installing a product
 - Must be a technical person
- RM 5.3 Acceptance tester
- Tests the functionality of new hardware and software
 - Takes the user perspective on whether the product does what it was intended to do
 - Liaises with the build developer (see [RM 5.2](#)) to agree test criteria and perform tests
 - Does not need to be technical
 - Must be familiar with the requirements of the product being tested
 - Is likely to be an end-user
- RM 5.4 Installer
- Installs new equipment in response to user requests
 - Must use the install build checklist and the appropriate build procedure to execute all installations
 - Liaises with build developer (see [RM 5.2](#)) for assistance if necessary
 - Refers any issues with build procedures to build developer or release manager (see [RM 5.1](#))
 - Will be a technical person
 - Will probably be involved in day-to-day Incident Management

RM 6 Review of Release Management

The purpose of this section is to help you review your implementation and ongoing operation of release management, check your understanding of the process, examine what a successful implementation should look like and consider what you have achieved by introducing it into your school. This will help you to assess how successful its introduction has been and point you back to the relevant sections in the Release Management process that you should revisit to make improvements, if these are necessary.

Start by reading the sections included in the recap of Release Management. When you have refreshed your memory and considered your own implementation alongside these descriptions, work through the checklist to identify any areas that you should revisit and perhaps re-implement or reinforce.

RM 6.1 Recap of Release Management

In Release Management we introduced the process of planning, building, testing and deploying ICT equipment, and managing software and software licences. We gave you an overview of the whole Release Management process and an implementation guide giving step-by-step instructions to help you implement a release management process that we believe is appropriate for the needs of schools. An operations guide gave you a list of ongoing activities required by the process in order for you to keep it going and reap the benefits. We described roles and responsibilities and offered guidance on how to assign roles. We removed anything non-essential to give you a lean process requiring the minimum of effort and resource.

Check your understanding of the process by going through [RM 6.1.1](#) to [RM 6.1.4](#) below.

RM 6.1.1 Release Management summary

Step	Tasks
Create a definitive software library.	Create a secure, central storage area for all software including: • original software • copies of software used for installation purposes. Keep a log of software licences and a record of which computers they have been assigned to.
Create standard builds for all hardware and software services.	Create a procedure document for each different service, including: • design details – the components that make up the service • installation details – the steps required to install the service correctly • stability test details – details of the environment the new service must work within without harming it • acceptance test details – functionality test criteria to ensure that the new service works correctly.

Step	Tasks
Make standard builds available to technical staff for installation as required.	Ensure that build instructions and the correct version of software are used for subsequent installations by: • storing the build software in the definitive software library • storing the build procedure in the configuration management database (CMDB) • informing all technical staff of the release.
Use standard builds for all new installations.	Use the installation checklist for every new hardware or software installation to ensure that: • staff take all of the appropriate steps and do not accidentally forget anything • end-user details are captured.
Ensure that new types of service go through the Release Management process before being installed.	Implement other FITS processes to draw attention to new requirements: • the incident/request process for new end-user requirements • the change management process for new infrastructure requirements.

RM

6.1.2

What you should expect now that you have implemented Release Management

- Technical support staff document all software and hardware installations.
- Technical support staff use build procedures and install checklists to install new equipment.
- End-users do not install their own hardware or software.
- No one installs software if no licence is available.
- The correct versions of software are always installed.
- Software and hardware services are stable.
- All new types of hardware and software are put through the Release Management process before they are deployed.

RM

6.1.3

What you should have achieved through Release Management

- You have a release policy describing the frequency and nature of upgrades.
- You receive fewer ad hoc requests for new hardware or software.
- All hardware and software installations have been tested and documented.
- Hardware and software is installed consistently each time.
- You spend less time resolving incidents and problems caused by badly installed hardware and software.
- All software is stored centrally.
- You can tell which version of software is the correct and current version.
- You have a list of all software licences.
- You know to whom or what each software licence is assigned.

Benefits of having implemented Release Management

- Planning for future releases is more efficient as a result of having a release policy.
- You handle changes to software more efficiently by bundling them together.
- End-users suffer less frequent disruption caused by changes.
- Having a repeatable process for installations is quicker and less error prone than relying on memory.
- Installing equipment in the same way each time makes support easier because you have to resolve incidents and problems only once.
- Emphasis on training before rollout means that users can make the most of new functionality as soon as it is available.
- Software version control ensures that no one reintroduces problems through installing the wrong version.
- New technical staff can follow documented instructions created by predecessors so continuity of skills is preserved.
- Technical staff unfamiliar with particular software or hardware have documented guidance which can help their personal development.
- Resolution of incidents is quicker through fast reinstallation of builds.
- Quicker resolution of problems is possible through comparison with standard build instructions.
- Less time spent on incidents and problems means more time available for proactive support.

Checklist

Use this checklist to identify any areas of Release Management that have not been entirely successful. Then reinforce them by revisiting and re-implementing the relevant section of the FITS process.

Characteristics of a successful implementation	FITS section to revisit if implementation has not yet been successful
You have assigned roles and responsibilities.	RM 3.2.1 Assigning roles and responsibilities in Release Management
All Release Management participants understand the process.	RM 2 Overview of Release Management
You have created a definitive software library.	RM 3.3 Step 2 Create definitive software library
You have created build procedures for all builds.	RM 3.3 Step 4 Create build
Build procedures and software are available to all installers.	RM 3.3 Step 5 Release build
All software and hardware installations are carried out in accordance with a checklist.	RM 3.3 Step 6 Install build
All software and hardware installation requirements are identified through the appropriate processes.	RM 4.1.1 Installing builds RM 4.2.1 When does it need doing? RM 4.3 Who does it?

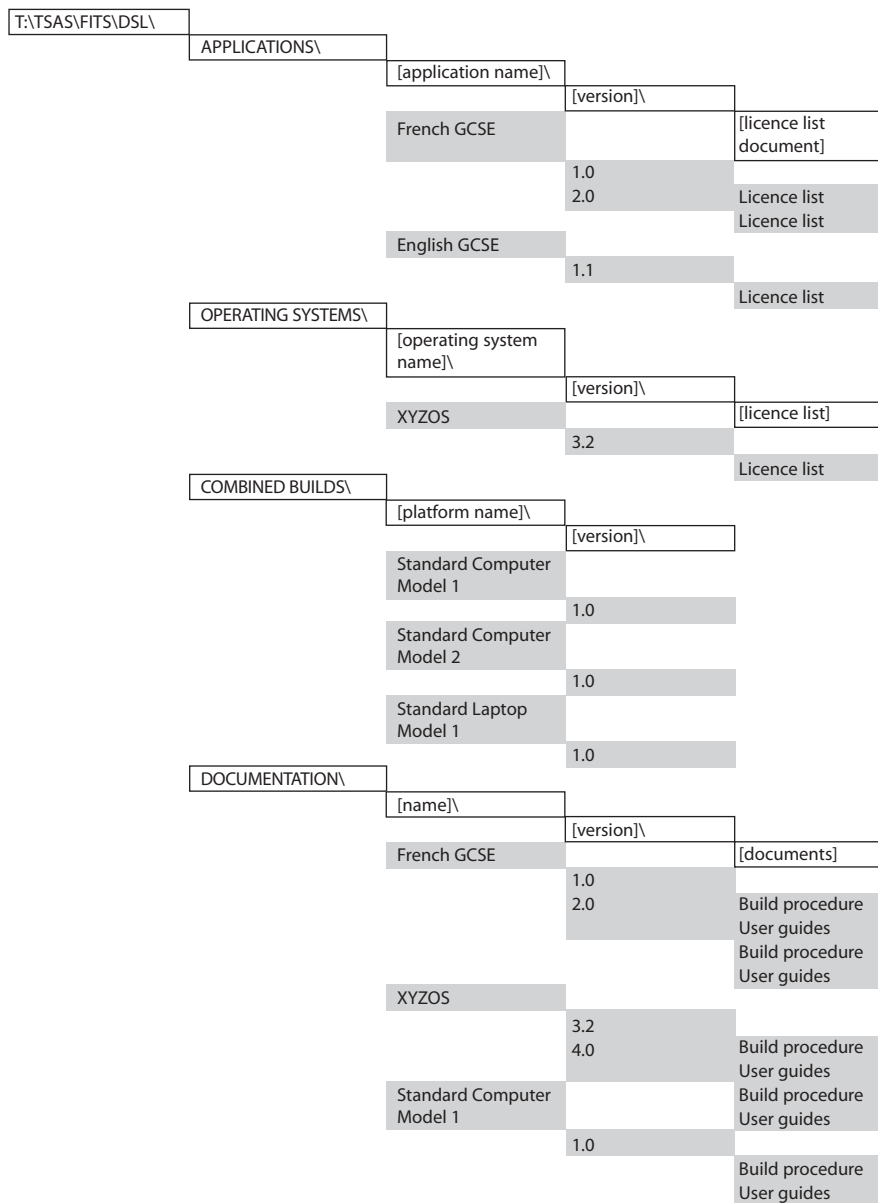
Characteristics of a successful implementation	FITS section to revisit if implementation has not yet been successful
You identify the need for new build procedures as they are required.	RM 4.1.2 Creating new builds RM 4.2.2 When does it need doing? RM 4.3 Who does it?
You produce release management reports regularly.	RM 4.1.3 Monitoring the Release Management process RM 4.2.3 When does it need doing? RM 4.3 Who does it?
You use release management reports to understand the use of the process, identify issues and make decisions.	RM 4.1.4 Making decisions about Release Management RM 4.2.4 When does it need doing? RM 4.3 Who does it?

If the above characteristics are all true of your school, congratulations on implementing a successful release management process! The next steps for you are to continue operating the process as described in [RM 4 Release Management Operations guide](#) and establish the process firmly. Work through this checklist at regular intervals to help you check that everyone responsible continues to carry out all aspects of the process. You can then refer to the relevant sections above to address any shortfalls as they arise.

Appendices

RM Appendix A Definitive software library (DSL) folder structure – example

Definitive software library example folder structure



Keep a separate list for each piece of software and store in the definitive software library (DSL) with the software itself

[illegible]

You can download the template from the FITS website
[<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

See FITS Release Management implementation guide for details on how to complete this form.

Build procedure – Hardware example

Name of Service: Standard desktop PC for all departments

Section 1: Design details

Describe the components and architecture of the service:

Component	Comments
Computer base unit	
Computer monitor	
Computer keyboard	
Mouse	
Network interface card	Check pre-installed in base unit
Operating System	
Network drivers	
Printer drivers	Check nearest shared printer for type
Standard email software	
Standard word processing software	

Section 2: Build details

Describe step-by-step instructions to install service:

No.	Description of step	Comments
1	Connect keyboard, mouse and monitor to base unit	
2	Plug in mains cable and power up computer	
3	Create desktop icons for email and word-processing	Note that email and word-processing software is pre-installed on computer when delivered
4	Configure printer driver	Select printer type nearest to end-user
5	Create network login ID	[firstname][lastname initial] e.g. traceyt
6	Create default password	Letmein
7	Set password expiry to next login	Forces password reset by end-user
8	Create email address	[firstname.lastname@domainname] e.g. tracey.tomlinson@hallcroft.ac.uk
9	Login as user and map shared drive	Shared area relates to user department

Section 3: Test details

Test stability and resolve issues of new service and impact on existing services:

No.	List all services tested	Pass	Comments
1	Operating system	✓	This is a basic computer build – there are no pre-existing services to test against, all services are new
2	Standard email tool	✓	
3	Standard word processing tool	✓	

You can download the template from the FITS website

[<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=releases&id=tt5297>].

Section 4: Acceptance test

Test functionality of new service:

No.	Test	Tester	Pass
1	Log in	Debbie Wiggins	✓
2	Open and send email	Debbie Wiggins	✓
3	Open word processing tool and create document	Debbie Wiggins	✓
4	Send document to printer	Debbie Wiggins	✓

Section 5: Store software in DSL

Make software available:

Name	Version	Path	✓
Operating system	2.2	T:\TSAS\FITS\DSL\Operating systems	✓
Standard email tool	4.3	T:\TSAS\FITS\DSL\Applications	✓
Standard word processing tool	3.1	T:\TSAS\FITS\DSL\Applications	✓

Section 6: Store documents in CMDB

Make documentation available:

Name	Version	Path	✓
Example build procedure – hardware.doc	1	T:\TSAS\FITS\Documentation\Examples	✓
ICT policy guide and user handbook	2	T:\TSAS\FITS\Documentation\General User Guides	✓
Email manual	1	ICT store cupboard in room 123	✓
Word processing manual	1	ICT store cupboard in room 123	✓

Section 7: Communicate availability

Inform installers of availability:

Role	Date	✓
[list all those responsible or authorised to install hardware or software, e.g.]		
Service desk	25/06/03	✓
Technicians	25/06/03	✓
Network manager	25/06/03	✓
ICT co-ordinator	25/06/03	✓
Suppliers	25/06/03	✓

This build was created by:	(build developer)	Date:	
-----------------------------------	-------------------	--------------	--

You can download the template from the FITS website
[<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

See FITS Release Management implementation guide for details on how to complete this form.

Build procedure – Software example

Name of Service: French GCSE software application

Section 1: Design details

Describe the components and architecture of the service:

Component	Comments
Software CD French GCSE	
Standard desktop computer	(or standard laptop computer)

Section 2: Build details

Describe step-by-step instructions to install service:

No.	Description of step	Comments
1	Boot up the computer or laptop	Login screen should appear
2	Log in	
3	Insert CD into CD drive	
4	Double click on My Computer icon	
5	Select CD drive	
6	Double click on setup.exe	
7	Follow set up instructions	
8	Select printer driver	Check printer local to computer (final) location and select accordingly
9	Overwrite rogue.dll with file dated 01/06/03 13:00	T:\TSAS\FITS\DSL\Operating Systems\XYZOS\1.0\DLL upgrades
10	Create icon on desktop	

Section 3: Test details

Test stability and resolve issues of new service and impact on existing services:

No.	List all services tested	Pass	Comments
1	French GCSE	✓	
2	Operating system	✓	
3	Standard email tool	✓	
4	Standard word processing tool	✓	Failed with rogue.dll supplied with French GCSE software. Upgrade required – see Section 2, Step 9.

You can download the template from the FITS website
 [<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

Section 4: Acceptance test

Test functionality of new service:

No.	Test	Tester	Pass
1	Log in	Paul Stonier	✓
2	Launch French GCSE	Paul Stonier	✓
3	Launch lesson 1 and follow interactive instructions	Paul Stonier	✓
4	Print results sheet	Paul Stonier	✓

Section 5: Store software in DSL

Make software available:

Name	Version	Path	✓
French GCSE	2.0	T:\TSAS\FITS\DSL\Applications	✓

Section 6: Store documents in CMDDB

Make documentation available:

Name	Version	Path	✓
French GCSE manual	2	ICT store cupboard in room 123	✓
French GCSE build procedure	1	T:\TSAS\FITS\Documentation\Build Procedures\	✓

Section 7: Communicate availability

Inform installers of availability:

Role	Date	✓
[list all those responsible or authorised to install hardware or software, e.g.]		
Service desk	25/06/03	✓
Technicians	25/06/03	✓
Network manager	25/06/03	✓
ICT co-ordinator	25/06/03	✓
Suppliers	25/06/03	✓

This build was created by:	(build developer)	Date:	
-----------------------------------	-------------------	--------------	--

You can download the template from the FITS website
[<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

Notification of new build procedure example

To: Service desk, ICT-co-ordinator, all ICT technicians
From: Tracey Tomlinson, ICT technician
Date: 30 June 2003

French GCSE software

This is to notify all ICT installers of the creation of a new build procedure:

Name of service:	French GCSE version 2
Location of procedure:	T:\TSAS\FITS\DSL\Documentation\French GCSE\2.0\
Date created:	26 June 2003
Build developer:	Tracey Tomlinson

This procedure should be used for all installations of this application.

If you require further information please contact me on extension 123.

Regards

Tracey Tomlinson

Tracey Tomlinson
ICT technician

See FITS Release Management implementation guide for details of how to complete this form.

Install build checklist example

Installer name:

Section 1: Install details

Gather and document required information:

What is required?	Standard computer	
Who is it for?	Name:	James Burke
	Location:	B Block
	Telephone:	Extension 2345
Has a build been released?	Yes → locate build procedure	
	No → create build procedure	
Is software licence available	Yes → assign licence in licence list	
	No → refer to purchasing authority	
When is the install required by?	End of Friday 27 June 2003	
How long will the install take?	2 hours	
Install scheduled for	Date:	Wednesday 25 June 2003
	Time:	14:00
Is training required?	Yes	
Training scheduled for	Date:	Wednesday 25 June 2003
	Time:	16:00
Trainer	Name:	Andrew Powell
Is user documentation available?	Yes → locate user documentation	

Section 2: Install checklist

Complete all tasks required:

Task	✓
Build procedure located	✓
User documentation located	✓
User contacted to schedule installation	✓
Installation scheduled	✓
Training scheduled	✓
Build installed	✓
All elements of build procedure followed	✓
Installation completed	✓
User informed of installation completion	✓
Training completed	✓
User documentation provided to user	✓
Additional user information provided to user	✓
Software licence list updated	✓
Incident/Request process completed	✓

Section 3: Install document details

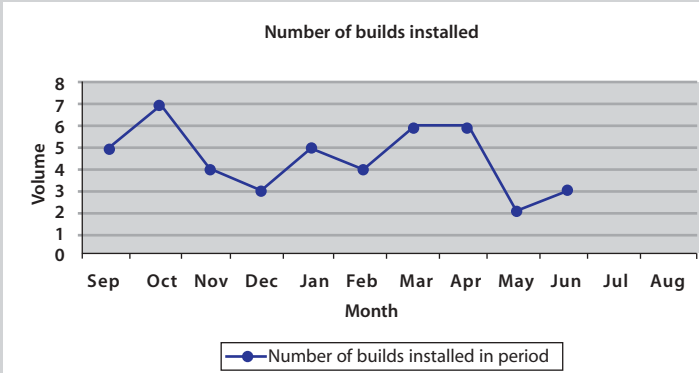
Store install build document for this install:

Name	Path	✓
Install checklist James Burke 1.doc	T:\TSAS\FITS\Documentation\Install Checklists	✓

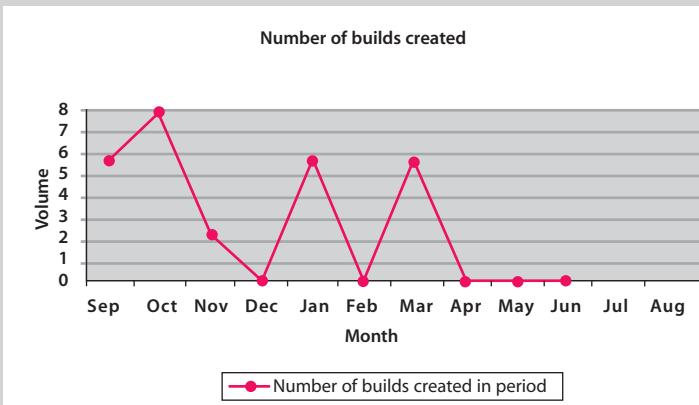
You can download the template from the FITS website
 [<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

Release Management report example

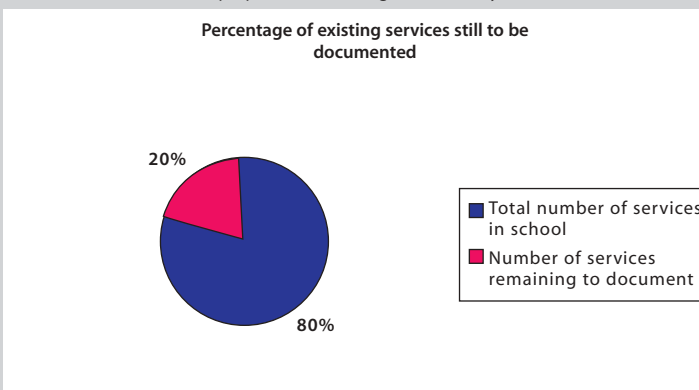
This shows the number of installs completed using build procedures in the given period



This shows the number of new build procedures created in the given period



This shows the proportion of existing services not yet documented



You can download the template from the FITS website
[<http://www.becta.org.uk/tsas/index.cfm?refsect=ntss&bcsect=default§=release&id=tt5297>].

Glossary

10Base-T	A networking standard that supports data transfer rates up to 100 Mbps (100 megabits per second). 10Base-T is based on the older Ethernet standard but is 10 times faster than Ethernet; it is often referred to as Fast Ethernet. Officially, the 10Base-T standard is IEEE 802.3u. Like Ethernet, 10Base-T is based on the CSMA/CD LAN access method.
AppleTalk	Inexpensive LAN (local area network) architecture built into all Apple Macintosh computers and laser printers. AppleTalk supports Apple's LocalTalk cabling scheme, as well as Ethernet and IBM Token Ring. It can connect Macintosh computers and printers, and even PCs if they are equipped with special AppleTalk hardware and software.
Asset	Component of a business process. Assets can include people, accommodation, computer systems, networks, paper records, fax machines, etc.
Availability	Ability of a component or service to perform its required function at a stated instant or over a stated period of time. It is usually expressed as the availability ratio: the proportion of time that the service is actually available for use by customers within the agreed service hours.
Availability Management	To ensure that ICT services are available for use consistently as agreed.
Bandwidth	The amount of data that can be transmitted in a fixed amount of time. For digital devices, the bandwidth is usually expressed in bits per second (bps).
Baseline	A snapshot or a position which is recorded. Although the position may be updated later, the baseline remains unchanged and available as a reference of the original state and as a comparison against the current position.
Bridge	A device that connects two LANs (local area networks), or two segments of the same LAN that use the same protocol, such as Ethernet or Token Ring.
Buffer	A temporary storage area, usually in RAM. The purpose of most buffers is to act as a holding area, enabling the CPU to manipulate data before transferring it to a device.
Build	The final stage in producing a usable configuration. The process involves taking one or more input configuration items and processing (building) them to create one or more output configuration items (eg software compile and load).
Capacity	Ability of available supply of processing power to match the demands made on it by the business, both now and in the future.
Capacity Management	To ensure that all ICT processing and storage capacity provision match present and evolving needs.
Category	Classification of a group of configuration items, change documents, incidents or problems.
Change	The addition, modification or removal of approved, supported or baselined hardware, network, software, application, environment, system, desktop build or associated documentation.

Change Management	The managed and recorded introduction of changes to hardware, software, services or documentation to minimise disruption to ICT operation and maintain accurate configuration information.
Client	The client part of a client/server architecture. Typically, a client is an application that runs on a personal computer or workstation and relies on a server to perform some operations. For example, an email client is an application that enables you to send and receive email.
Client/server architecture	A network architecture in which each computer or process on the network is either a client or a server. Servers are powerful computers or processes dedicated to managing disk drives (file servers), printers (print servers) or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients rely on servers for resources such as files, devices and even processing power.
Configuration management database (CMDB)	A database which contains all relevant details of each ICT asset, otherwise known as a configuration item (CI), and details of the important relationships between CIs.
Configuration Management	Implementing and maintaining up-to-date records of ICT hardware, software, services and documentation, and showing the relationships between them.
Definitive software library (DSL)	The library in which the definitive authorised versions of all software CIs are stored and protected. It is a physical library or storage repository where master copies of software versions are placed. This one logical storage area may in reality consist of one or more physical software libraries or filestores. They should be separate from development and test filestore areas. The DSL may also include a physical store (fire-proof safe, for example) to hold master copies of bought-in software. Only authorised software, strictly controlled by Change Management and Release Management, should be accepted into the DSL. The DSL exists not directly because of the needs of the Configuration Management process, but as a common base for the Release Management and Configuration Management processes.
Device	Any computer or component that attaches to a network.
Error trap	A signal informing a program that an event has occurred. When a program receives an interrupt signal, it takes a specified action (which can be to ignore the signal). Interrupt signals can cause a program to suspend itself temporarily to service the interrupt.
Ethernet	A LAN (local area network) architecture developed in 1976 by Xerox Corporation in co-operation with DEC and Intel. Ethernet uses a bus or star topology and supports data transfer rates of 10 Mbps. The Ethernet specification served as the basis for the IEEE 802.3 standard, which specifies the physical and lower software layers. Ethernet is one of the most widely implemented LAN standards.
FDDI (Fibre Distributed Data Interface)	A set of ANSI protocols for sending digital data over fibre optic cable. FDDI networks are token-passing networks, and support data rates of up to 100 Mbps (100 million bits) per second. FDDI networks are typically used as backbones for wide area networks.
Financial Management	To ensure that the ICT and technical resources are implemented and managed in a cost-effective way.

Firewall	A system designed to prevent unauthorised access to or from a private network. Firewalls can be implemented in both hardware and software, or a combination of both. Firewalls are frequently used to prevent unauthorised internet users from accessing private networks connected to the internet, especially intranets. All messages entering or leaving the intranet pass through the firewall, which examines each message and blocks those that do not meet the specified security criteria.
Gateway	A node on a network that serves as an entrance to another network. In schools, the gateway is the computer that routes the traffic from a workstation to the outside network that is serving web pages. In homes, the gateway is the ISP that connects the user to the internet.
Gigabit	When used to describe data transfer rates, it refers to 10 to the 9th power (1,000,000,000) bits. Gigabit is abbreviated Gb, as opposed to gigabyte, which is abbreviated GB.
HTTP (hypertext transfer protocol)	The underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the web server directing it to fetch and transmit the requested web page.
Hub	A connection point for devices in a network. Hubs are commonly used to connect segments of a LAN (local area network). A hub contains multiple ports. When a packet arrives at one port, it is copied to the other ports so that all segments of the LAN can see all packets.
ICT	The convergence of information technology, telecommunications and data networking technologies into a single technology.
Incident	Any event which is not part of the standard operation of a service and which causes, or may cause, an interruption to, or a reduction in, the quality of that service.
Incident Management	To detect, diagnose and resolve ICT incidents as quickly as possible and minimise their adverse impact on normal operation.
ITIL	The OGC IT Infrastructure Library – a set of guides on the management and provision of operational IT services.
LAN	A computer network that spans a relatively small area. Most local area networks (LANs) are confined to a single building or group of buildings.
LocalTalk	The cabling scheme supported by the AppleTalk network protocol for Macintosh computers. Most local area networks that use AppleTalk, such as TOPS, also conform to the LocalTalk cable system. Such networks are sometimes called LocalTalk networks.
Logical topology	The logical topology is the way that the signals act on the network media, or the way that the data passes through the network from one device to the next without regard to the physical interconnection of the devices.
MAC (media access control) address	Each device on a network can be identified by its MAC address, a hardware address that uniquely identifies each node of a network. In IEEE 802 networks, the data link control (DLC) layer of the OSI reference model is divided into two sub-layers: the logical link control (LLC) layer and the MAC layer. The MAC layer interfaces directly with the network media. Consequently, each different type of network media requires a different MAC layer.

Management information base (MIB)	A management information base (MIB) is a database of objects that can be monitored by a network management system. Both SNMP and RMON use standardised MIB formats that allow any SNMP and RMON tools to monitor any device defined by a MIB.
Network	A group of two or more computer systems linked together. The two types of computer networks of interest to schools are LANs (local area networks) and WANs (wide area networks).
Network interface card (NIC)	A network interface card (NIC) is an expansion board inserted or built into a computer so that the computer can be connected to a network. Most NICs are designed for a particular type of network, protocol, although some can serve multiple networks.
Network traffic	The load on a communications device or system.
Node	A processing location. A node can be a workstation or some other device, such as a printer. Every node has a unique network address, sometimes called a data link control (DLC) address or media access control (MAC) address.
OSI reference model	The OSI (open system interconnection) model defines a networking framework for implementing protocols in seven layers. Control is passed from one layer to the next, starting at the application layer in one station, and proceeding to the bottom layer, over the channel to the next station, and back up the hierarchy.
Packet	A piece of a message transmitted over a packet-switching network. One of the key features of a packet is that it contains the destination address in addition to the data.
Packet switching	Refers to protocols in which messages are divided into packets before they are sent. Each packet is then transmitted individually and can even follow different routes to its destination. Once all the packets forming a message arrive at the destination, they are recompiled into the original message.
Peer-to-peer network	A type of network in which each workstation has equivalent capabilities and responsibilities. This differs from client/server architectures, in which some computers are dedicated to serving the others.
Physical topology	The physical layout of devices on a network. Every LAN (local area network) has a topology – the way the devices on a network are arranged and how they communicate with each other.
Port	In TCP/IP and UDP networks, an endpoint to a logical connection. The port number identifies what type of port it is. For example, port 80 is used for HTTP traffic.
Problem	The underlying cause of an incident or incidents.
Problem Management	The detection of the underlying causes of incidents and their resolution and prevention.
Protocol	An agreed format for transmitting data between two devices.
Protocol stack	A set of network protocol layers that work together. The OSI reference model that defines seven protocol layers is often called a stack, as is the set of TCP/IP protocols that define communication over the internet.

Proxy server	A server that sits between a client application, such as a web browser, and a real server. It intercepts all requests to the real server to see if it can fulfil the requests itself. If not, it forwards the request to the real server.
Release Management	To plan, test and manage the successful implementation of software and hardware. To define release policy and to ensure that master copies of all software are secured centrally.
Remote monitoring (RMON)	Remote monitoring (RMON) is a network management protocol that allows network information to be gathered at a single workstation. For RMON to work, network devices such as hubs and switches must be designed to support it.
Request for change	Form or screen used to record details of a request for a change to any CI within an infrastructure, or to procedures and items associated with the infrastructure.
Router	A device that forwards data packets along networks. A router is connected to at least two networks, commonly two LANs (local area networks) or WANs (wide area networks) or a LAN and its ISP's network. Routers are located at gateways, the places where two or more networks connect.
Segment	A section of a network that is bounded by bridges, routers or switches. Dividing an Ethernet into multiple segments is one of the most common ways of increasing available bandwidth on the LAN.
Server	A workstation or device on a network that manages network resources. For example, a file server is a computer and storage device dedicated to storing files. Any user on the network can store files on the server. A print server is a computer that manages one or more printers, and a network server is a computer that manages network traffic. A database server is a computer system that processes database queries.
Service Continuity Management	To minimise the impact on ICT service of an environmental disaster and put in place and communicate a plan for recovery.
Service Desk	The single point of contact within the school for all users of ICT and the services provided by Technical Support.
Service level agreement	Written agreement between a service provider and the customer(s) that documents agreed service levels for a service.
Service Level Management	The process of defining, agreeing and documenting required service levels and ensuring that these levels are met.
Simple network management protocol (SNMP)	A set of protocols for managing complex networks. SNMP works by sending messages, called protocol data units (PDUs), to different parts of a network. SNMP-compliant devices, called agents, store data about themselves in management information bases (MIBs) and return this data to the SNMP requesters.
Star topology	A LAN (local area network) that uses a star topology in which all nodes are connected to a central computer. The main advantages of a star network are that one malfunctioning node does not affect the rest of the network and that it is easy to add and remove nodes.
Switch	A device that filters and forwards packets between segments of a LAN (local area network). Switches operate at the data link layer (layer 2) and sometimes the network layer (layer 3) of the OSI reference model and therefore support any packet protocol.

TCP/IP (Transmission Control Protocol/Internet Protocol)	The suite of communications protocols used to connect hosts on the internet. TCP/IP uses several protocols, the two main ones being TCP and IP.
Token ring	A type of computer network in which all the computers are arranged (schematically) in a circle. A token, which is a special bit pattern, travels around the circle. To send a message, a computer catches the token, attaches a message to it, and then lets it continue to travel around the network.
Topology	The shape of a LAN (local area network) or other communications system. Topologies are either physical or logical.
User datagram protocol (UDP)	A connectionless protocol that, like TCP, runs on top of IP networks. Unlike TCP/IP, UDP/IP provides very few error recovery services, offering instead a direct way to send and receive datagrams over an IP network. It is used primarily for broadcasting messages over a network.
WAN	A computer network that spans a relatively large geographical area. Typically, a wide area network (WAN) consists of two or more LANs (local area networks). Computers connected to a wide area network are often connected through public networks, such as the telephone system. They can also be connected through leased lines or satellites. The largest WAN in existence is the internet.
Workstation	Any computer connected to a LAN (local area network).