

Summary

Software change control is both a managerial as well as technical activity and is essential for proper software quality control. At the project level these activities should be included as part of the project plan or in a software change control plan.

CC.C provides a documentation template for the software change control plan.

Change control procedures cover the establishment of methods for identifying, storing and changing system items that pass through development, integration, implementation and operations.

Tools for software change control are widely available and their use is strongly recommended to ensure that the procedures are applied consistently and efficiently.

CC.D provides a discussion of software change control tools in terms of their capabilities.

CC.1 Configuration Item Identification

1 | D O M A | CC.1.1 Define Configuration Items (CIs)

1 | D O M A | CC.1.1.1

If the software system is in its development phases of the software life cycle, the Software Project Manager (SPM) or delegate (Change Manager) shall define the CIs and the conventions for their identification for software change control.

During the operations and maintenance phases of the software life cycle, the Software Maintenance Manager or delegate (Change Manager) is responsible for understanding and using the CIs and the conventions for their identification, which have been previously defined in the development phases of the software life cycle.

1 | D O M A | CC.1.1.2

The Change Manager should reference the project plan, software requirements document and the design specifications in defining the CIs that are required to be managed and their hierarchy including:

- a) requirements, designs, code, test specifications;

CC.A provides guidelines on what configuration items to include for software change control.

- b) system documentation, user documentation;
- c) project plans, test plans;
- d) standards and procedures.

1 | D O M A CC.1.2 Define Configuration Identification Conventions

1 | D O M A CC.1.2.1

The Change Manager should ensure that the configuration identification conventions used should state how to:

- a) Identify a CI;
- b) Decide who is the control authority of a CI;
- c) Describe the history of a CI.

1 | D O M A CC.1.2.2

The Change Manager should follow specific rules in uniquely identifying CIs including:

- a) Each CI should be identified uniquely through its name, type and version attributes;
- b) Documents should have standard names;
- c) Software items should be named after the software components;
- d) The type field should identify the processing the CI is intended for;
- e) Different versions of CIs should be given different version numbers;
- f) The configuration identification method used must be capable of accommodating new CIs without requiring modification to the identifiers of existing CIs.

1 | D O M A CC.1.2.3

The Change Manager should follow specific rules in deciding who is the control authority to decide what changes will be made to a CI. In a software project for

example:

- a) Software authors (programmers and document writers) are the control authorities for low-level CI such as documents and code;
- b) Software Project Managers (SPMs) or the Software Maintenance Managers (SMMs) supported by the System Librarian (SL) are the control authorities of high level CIs i.e. baselines and releases.

1 D O M L CC.1.2.4

For large organizations and/or projects, a review group is used in place of the SPM and SMM to control changes to baselines. Representation in the review groups could comprise of the SPM/SMM, the SL, user representatives, team leaders and quality assurance personnel.

1 D O M A CC.1.2.5

The Change Manager should follow specific rules in describing the history of a CI including:

- a) The development history of a CI should be recorded from the moment it is first submitted for review or integration;
- b) For documents, a history of changes should be provided;
- c) For source code, the development history should be stored in the module header tying them back to the software change requests that actioned them.

CC.B provides guidelines on how to identify CIs, their control authority and to provide for history descriptions.

CC.2 Change Control Environment

1 D O M A CC.2.1 Establish a Secure CI Environment

1 D O M A CC.2.1.1

To ensure that software is never lost or its integrity compromised, the Change Manager must ensure that a secure environment is established for:

- a) storing all CIs,
- b) for maintaining safely storage media e.g. paper, magnetic disk.

1 | D O M A CC.2.2 Define CI System Libraries**1 | D O M A** CC.2.2.1

At the very minimum, the following software libraries are required to store CIs:

- a) development (or dynamic) libraries to store CIs that are being written or unit tested (under the control of individual developers);
- b) test (or master) libraries, where current baseline CIs are stored and integration and systems testing carried out (under the control of the Change Manager);
- c) production (or archive) libraries where releases or retired baselines are stored (under the control of the Change Manager).

1 | D O M A CC.2.2.2

The Change Manager should ensure that rules are established and followed with regard to the development and maintenance of system libraries including:

- a) Development libraries should be created and maintained by the software authors in their own work place.
- b) Test and production libraries should be created and maintained by the System Librarian who should report to the Change Manager with responsibilities for:
 - establishing new test libraries;
 - updating test libraries;
 - backup of test libraries to production libraries;
 - control of access to test and production libraries.

1 | D O M A CC.2.2.3

The Change Manager should ensure that rules are established and followed with regard to access to system libraries that make it impossible:

- a) to access CIs without correct authorization;
- b) for two people to simultaneously update the same CI.

Tables 1 and 2 below provide the software access rights that should be granted development staff and system librarians.

Table 1: Development Staff Access Rights

Library \ Access Right	Read	Insert	Replace	Delete
Development Libraries	Yes	Yes	Yes	Yes
Test Libraries	Yes	No	No	No
Production Libraries	Yes	No	No	No

Table 2: Software Librarian Access Rights

Library \ Access Right	Read	Insert	Replace	Delete
Development Libraries	Yes	No	No	No
Test Libraries	Yes	Yes	Yes	Yes
Production Libraries	Yes	Yes	Yes	Yes

Legend: Read access right allows a CI to be examined or copied.
 Insert access right allows a new CI to be added to a library.
 Delete access right allows a CI to be removed from a library.
 Replace access right allows a CI to be removed from the library and another version inserted.

1 | D O M A | CC.2.3 Control Media

1 | D O M A | CC.2.3.1

The Change Manager must ensure that all CIs are stored on media that can be controlled i.e. recover from media loss quickly (e.g. tapes, disks).

1 | D O M A | CC.2.3.2

The Change Manager should ensure that all CI media should be labeled visibly (e.g. sticky label) and electronically (e.g. tape header) with a standard format containing:

- a) Project name;
- b) CI identifier (name, type, version);
- c) Date of storage;
- d) Content description.

1 | D O M A | CC.2.3.3

In addition, the Systems Librarian should ensure that regular backups of development, master and archive libraries are carried out as standard procedure.

CC.3 Change Request & Execution

Summary

Changes to CIs are normal in the evolution of a software system. CI changes could be the result of defects, change requests or deficiencies identified during the software life-cycle e.g. systems testing, acceptance testing, operations etc., by users development staff and quality assurance personnel.

The common procedures that are followed when CIs are changed include:

- an analysis of the problem descriptions and/or change requests;
- a decision on which controlled CIs are to be changed;
- retrieval from the test library of CIs that are required to be changed;
- return of changed CIs to the master library for storage;
- re-verification/re-testing of all changed CIs.

1 D O M A CC.3.1 Request Change

1 D O M A CC.3.1.1

During a software life cycle, the originator of a change (normally users, development staff and quality assurance personnel) prepares a software problem report (SPR) to contain:

CC.E provides a form template for documenting the SPR

- a) CI name;
- b) CI version number;
- c) Priority of problem with respect to other problems;
- d) A description of the problem;
- e) Operating environment;
- f) Recommended solution (where possible).

1 D O M A CC.3.1.2

The software project manager (SPM) or software maintenance manager (SMM) assigns the SPR to an expert (normally a development or maintenance staff) for diagnosis.

1 D O M A CC.3.1.3

After diagnosis, the expert must prepare a software change request (SCR) if modifications to the software and/or documentation are required to include:

- a) CI title or name;
- b) CI version or release number;
- c) Priority of problem with respect to other problem;
- d) Changes including documentation required;
- e) Responsible staff;
- f) Estimate start & end date and manpower effort.

1 D O M A CC.3.1.4

The SPR and associated SCRs are then handed over to the SPM or SMM for evaluation and decision on whether to:

- a) Close the SPR because the update is completed;
- b) Change the software and/or documentation on the basis of recommendations in the associated SCR;
- c) Reject the SPR;
- d) Define an action that needs to be taken before a decision to close, change or reject the SPR.

2 D O M L CC.3.1.5

In large organizations, the evaluations and decisions should be made by a software review group normally comprising the SPM or SMM, user representatives and software quality personnel.

1 D O M A CC.3.1.6

Approved SCRs are passed to the software developers or maintainers to implement the changes and produce the software modification report (SMR) for each SCR.

CC.F and CC.G provide form templates for preparing the SCR and SMR respectively.

1 D O M A CC.3.1.7

The software developers or maintainers should ensure that all changes to code and/or documentation are made in

development libraries, testing carried out in test libraries, and tested versions released to production libraries.

1 D O M A CC.3.1.8

The software developers or maintainers request retrieval from the System Librarian of the code and/or documentation to be changed from the production library, modifies the code and/or documentation (including User Requirements Document) in the development libraries and requests return from the System Librarian of the changed code and/or documentation back to the test library for any further testing required. Tested versions will then be passed to the production libraries for release.

Refer User Requirements Management practice (Section UR.5 Requirements Management) on how software changes are kept in step with user requirements and vice versa.

1 D O M A CC.3.2 Status Reporting

1 D O M A CC.3.2.1

The Systems Librarian should keep *track of the status* of each and every CI to include:

- a) A description of whether the CI has passed project milestones such as unit testing and systems testing, providing management with visibility of product development;
- b) A record of the status of every SPR, SCR and SMR related to every CI in the baseline (Table 3 shows how such records might be kept).

Table 3: Example of Configuration Item Status

CI Name Version	SPR	Date Submitted	Review Date	SPR Decision	Related SCR	Related SMR	Completi on Date
LOADER V1.1	2	5/8/98	1/9/98	Update	2	2	1/10/98
LOADER V1.1	3	7/8/98	1/9/98	Update	3	2	1/10/98
EDITOR V1.2	4	9/8/98	1/9/98	Rejected	-	-	-
MERGER V1.0	1	1/8/98	1/9/98	Update	1	1	3/10/98

Legend: V = version

1 D O M A CC.3.3 Code Release

1 | D O M A CC.3.3.1

The software developer or maintainer should perform regression testing (e.g. unit, integration and system testing) for all software code and/or documentation that have been modified before their release.

1 | D O M A CC.3.3.2

After successful regression testing of software code and/or documentation, the software developer or maintainer should complete a Software Release Note (SRN) which contains:

- a) An overview of the release;
- b) A CI list that catalogues the CIs in the release;
- c) The problems that have been addressed and/or new requirements incorporated;
- d) Installation instructions.

Appendix CC.H provides a form template of a SRN.

1 | D O M A CC.3.3.3

The software developer or maintainer should request release by submitting the completed SRN to the Change Manager who checks that:

- a) Release documentation is complete;
- b) For each release, documentation and code are consistent;
- c) Releases should be self-identifying (e.g. labels, printed output etc. provided) so that users know what the software is.

1 | D O M A CC.3.3.4

After successful checking, the Change Manager then instructs the System Librarian to release the code and/or documentation providing him with all the release documentation and ensuring that:

- a) Old releases are retained for reference;
- b) Where possible, the previous release should be kept

online during the change-over period, to allow comparisons and as a fallback.

CC.4 Change Notification

1 D O M L CC.4.1 Provide Change Notification & Status Reports

1 D O M L CC.4.1.1

As and when changes have been successfully made, the system librarian should provide notification of the change to the originators of the change (normally users, development staff and quality assurance personnel) and those impacted by the changes e.g. operation and development staff.

1 D O M L CC.4.1.2

On a regular (e.g. weekly) and on an on request basis (requests from software project/maintenance managers, users and developers), the system librarian should provide reports of the status of CIs including highlights of:

- a) Contents, dates and version/issue of each baseline and associated CI (refer Table 4 for an example of baseline records);
- b) Status of problem report and their solutions;
- c) Status of every SPR, SCR and SMR related to every CI in the baseline.

Table 4: Highlights of Baseline Records Showing Issue and Revision Numbers of each Document and Version Numbers of the Code

CI \ Baseline	URD	SRD	DS	Coding	Testing	Acceptance
Software Project Plan	1.0	2.0	3.0	4.0	4.1	4.2
Program A				1.0	1.4	1.5
Program B				1.0	1.3	1.4
Program C				1.0	1.4	1.4
Compiler				5.2	5.2	5.2
Operating System				6.1	6.1	6.1

Legend: **URD** = User Requirements Document
SRD = Systems Requirements Document
DS = Design Specifications