IEEE STANDARDS ASSOCIATION

IEEE Standard for Configuration Management in Systems and Software Engineering

IEEE Computer Society

Sponsored by the
Software & Systems Engineering Standards Committee

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

16 March 2012

IEEE Std 828™-2012
(Revision of
IEEE Std 828-2005)

IEEE Std 828™-2012
(Revision of
IEEE Std 828-2005)

IEEE Standard for Configuration Management in Systems and Software Engineering

Sponsor

Software & Systems Engineering Standards Committee
of the
IEEE Computer Society

Approved 6 February 2012

IEEE-SA Standards Board

Abstract: This standard establishes the minimum requirements for processes for Configuration Management (CM) in systems and software engineering. The application of this standard applies to any form, class, or type of software or system. This revision of the standard expands the previous version to explain CM, including identifying and acquiring configuration items, controlling changes, reporting the status of configuration items, as well as software builds and release engineering. Its predecessor defined only the contents of a software configuration management plan. This standard addresses what CM activities are to be done, when they are to happen in the life cycle, and what planning and resources are required. It also describes the content areas for a CM Plan. The standard supports ISO/IEC/IEEE 12207:2008 and ISO/IEC/IEEE 15288:2008 and adheres to the terminology in ISO/IEC/IEEE Std 24765 and the information item requirements of IEEE Std 15939™.
Keywords: change control, configuration accounting, configuration audit, configuration item, IEEE 828, release engineering, software builds, software configuration management, system configuration management

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: ISBN 978-0-7381-7232-3 STD97219
Print: ISBN 978-0-7381-7246-0 STDPD97219

Notice and Disclaimer of Liability Concerning the Use of IEEE Documents: IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

Use of an IEEE Standard is wholly voluntary. IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon any IEEE Standard document.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

Translations: The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official Statements: A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on Standards: Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important to ensure that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. Any person who would like to participate in evaluating comments or revisions to an IEEE standard is welcome to join the relevant IEEE working group at http://standards.ieee.org/develop/wg/.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854
USA

Photocopies: Authorization to photocopy portions of any individual standard for internal or personal use is granted by The Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Notice to users

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://standards.ieee.org/index.html or contact the IEEE at the address listed previously. For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA Website at http://standards.ieee.org/index.html.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website http://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this IEEE standard was completed, the Configuration Management Working Group had the following membership:

Chuck Walrad, *Chair*
Mike Smith, *Vice Chair*
Diego Pamio, *Secretary*
Ranata Johnson, *Editor*

Bob Aiello
Adonica Geiger
Darrel Strom
Christopher Ward
Ben Weatherall
Bernhard Westfechtel
M. Karen Woolf

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Ed Addario
Johann Amsenga
Chris Bagge
Bakul Banerjee
Charles Barest
Juris Borzovs
Pieter Botman
Lyle Bullock
Lawrence Catchpole
Keith Chow
Paul Croll
Geoffrey Darnton
Thomas Dineen
Jennifer Doman
Sourav Dutta
Harriet Feldman
Andrew Fieldsend
Eva Freund
David Friscia
David Fuschi
George Gianacakes
Gregg Giesler
Lewis Gray
Ron Greenthaler
Randall Groves
John Harauz
Werner Hoelzl
Robert Holibaugh
Bernard Homes
Peter Hung
Atsushi Ito
Mark Jaeger
Cheryl Jones
Hirofumi Kamibayashiyama
Piotr Karocki
Stanley Klein
Dwayne Knirk
Thomas Kurihara
George Kyle
Susan Land
Kenneth Lang
David Leciston
Vincent Lipsio
Greg Luri
Wayne W. Manges
Edward McCall
Avygdor Moise
James Moore
Adi Mulawarman
Michael S. Newman
Chris Osterloh
Mark Paulk
Robert Peterson
William Petit
Annette Reilly
Robert Robinson
Fernando Lucas Rodriguez
Garry Roedler
Randall Safier
Helmut Sandmayr
Bartien Sayogo
Robert Schaaf
David Schultz
Stephen Schwarm
Gil Shultz
Carl Singer
David Singleton
Michael Smith
Kapil Sood
Friedrich Stallinger
Thomas Starai
Darrel Strom
Walter Struppler
Gerald Stueve
March Stutzman
K. Subrahmanyam
Richard Thayer
Thomas Tullia
John Vergis
Charlene Walrad
M. Karen Woolf
Jian Yu
Oren Yuen
Janusz Zalewski
Shuhui Zhang

When the IEEE-SA Standards Board approved this standard on 6 February 2012, it had the following membership:

Richard H. Hulett, *Chair*
John Kulick, *Vice Chair*
Robert M. Grow, *Past President*
Judith Gorman, *Secretary*

Masayuki Ariyoshi	Jim Hughes	Jon Walter Rosdahl
William Bartley	Joseph L. Koepfinger*	Sam Sciacca
Ted Burse	David J. Law	Mike Seavey
Clint Chaplin	Thomas Lee	Curtis Siller
Wael Diab	Hung Ling	Phil Winston
Jean-Philippe Faure	Oleg Logvinov	Howard L. Wolfman
Alexander Gelman	Ted Olsen	Don Wright
Paul Houzé	Gary Robinson	

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*
Satish Aggarwal, *NRC Representative*

Julie Alessi
IEEE Standards Program Manager, Document Development

Malia Zaman
IEEE Standards Program Manager, Technical Program Development

Introduction

This introduction is not part of IEEE Std 828-2012, IEEE Standard for Configuration Management in Systems and Software Engineering.

This revision to IEEE Std 828™-2005 replaces the earlier focus on the contents of a Software Configuration Management Plan (SCMP) with a focus on the processes that comprise System and Software Configuration Management. The standard for the SCMP is now included as a normative Annex.

Configuration Management in Systems and Software Engineering is a specialty discipline within the larger discipline of Configuration Management (CM). The purpose of Configuration Management is to:

a) Identify and document the functional and physical characteristics of any product, component, result, or service

b) Control any changes to such characteristics

c) Record and report each change and its implementation status

d) Support the audit of the products, results, services, or components to verify conformance to requirements

Configuration Management is essential to Systems Engineering and to Software Engineering.

CM establishes and protects the integrity of a product or product component throughout its lifespan, from determination of the intended users' needs and definition of product requirements through the processes of development, testing, and delivery of the product, as well as during its installation, operation, maintenance, and eventual retirement. In so doing, CM processes interface with all other processes involved in the product's life.

Annex A provides a condensed view of the purposes and outcomes of the lower level CM processes described in this standard.

Contents

IEEE Standard for Configuration Management in Systems and Software Engineering

1. Overview

1.1 Scope

This standard establishes the minimum requirements for Configuration Management (CM) in Systems and Software Engineering, without restriction to any form, class, or type.

1.2 Purpose

This standard describes CM processes to be established, how they are to be accomplished, who is responsible for doing specific activities, when they are to happen, and what specific resources are required. It addresses CM activities over a product's life cycle. This standard is consistent with IEEE's Software Engineering Body of Knowledge (SWEBOK), ISO/IEC/IEEE 12207:2008 and ISO/IEC/IEEE 15288:2008.

2. Definitions, acronyms, and abbreviations

For the purposes of this document, the following terms and definitions apply. ISO/IEC/IEEE 24765[1] and the *IEEE Standards Dictionary: Glossary of Terms and Definitions*[2] should be consulted for terms not defined in this clause.

2.1 Definitions

baseline: (1) specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures (2) formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle. (ISO/IEC 24765:2009)

NOTE—A software baseline is a set (one or more) of software configuration items formally designated and fixed at a specific time during the software life cycle. A baseline, together with all approved changes to the baseline, represents the current approved configuration. The term is thus used to refer to a particular version of a software configuration item that has been agreed on, e.g., as a stable base for further development or to mark a specific project milestone. In either case, any new baseline is agreed through the project's agreed change control procedures.[3]

build: (n) an operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide (ISO/IEC 24765:2009); (v) to perform the steps required to produce an instance of the product.

NOTE—In software, this means processing source files to derive target files. In hardware, this means assembling a physical object.

configuration control board (CCB): (1) a group of people responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of approved changes (ISO/IEC 24765:2009) (2) qualified personnel who evaluate, for approval or disapproval, all proposed changes to the current developmental baseline. (ISO/IEC 2382-20:1990, 20.07.08) Syn: change control board (ISO/IEC 24765:2009)

configuration item (CI): aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process. (ISO/IEC 24765:2009)

NOTE—Configuration items may vary widely in complexity, size and type, ranging from an entire system including all hardware, software and documentation, to a single module or a minor hardware component.

configuration management (CM): (1) a discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements (ISO/IEC 24765:2009) (2) technical and organizational activities comprising configuration identification, control, status accounting, and auditing. (ISO/IEC 29881:2008--FiSMA 1.1 functional size measurement method, 4.9)

constituent configuration item: an individual item to be controlled that is a constituent (part) of a larger configuration item, such as a reference model, hardware prototype or software build.

[1] The database standard ISO/IEC/IEEE 24765, Systems and Software Engineering-Vocabulary can be accessed at www.computer.org/sevocab.
[2] *IEEE Standards Dictionary: Glossary of Terms and Definitions* is available at http://shop.ieee.org.
[3] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

configuration management authority: Any person(s) or group designated to be responsible for assuring that CM activities are planned and carried out.

configuration management database (CMDB): a specific type of repository for CM information, usually a data store, used to record attributes of configuration items, and the relationships between configuration items, throughout their lifecycle. (ISO/IEC 20000-1:2011)

NOTE 1—The requirements for the specific type of CMDB tool used for a given project depend on the size and complexity of the project and product. For a small-scale project, a CMDB can be as simple as a spreadsheet.

NOTE 2—Some commercial product CMDBs are specific to CIs in physical configurations, such as ports opened in a firewall; software CMDBs are inherent to CM tools and enable identification of all CIs in all states. CMDBs may be hierarchical or federated as needed to accurately determine and organize information about configuration items. CMDBs may provide information to help keep the Configuration Management System (CMS) updated and accurate.

functional configuration audit: an audit conducted to verify that the development of a configuration item has been completed satisfactorily, that the item has achieved the performance and functional characteristics specified in the functional or allocated configuration identification, and that it is operational and support documents are complete and satisfactory. (ISO/IEC 24765:2009)

ISO file: file "image" of an entire CD or DVD that is encoded according to ISO 9660.

life cycle: evolution of a system, product, service, project, or other human-made entity from conception through retirement. (ISO/IEC/IEEE 12207:2008)

physical configuration audit: an audit conducted to verify that a configuration item, as built, conforms to the technical documentation that defines it. (ISO/IEC 24765:2009)

NOTE—In addition to the definitions given in ISO/IEC/IEEE 24765, the following explanation applies: For software, the purpose of the software physical configuration audit (PCA) is to ensure that the design and reference documentation is consistent with the as-built software product.

release: (1) a delivered version of an application that may include all or part of an application (2) collection of new and/or changed configuration items that are tested and introduced into the live environment together (3) a software version that is made formally available to a wider community. (ISO/IEC 24765:2009)

release plan: a plan that describes what portions of system functionality will be implemented in which releases and the rationale for each release. It includes or provides reference to a description of release contents, release schedule, release impacts and release notifications.

repository: a (1) a collection of all software-related artifacts belonging to a system (2) the location/format in which such a collection is stored. (ISO/IEC 24765:2009)

software: (1) all or part of the programs, procedures, rules, and associated documentation of an information processing system. (2) computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. (ISO/IEC 24765:2009)

software item: (1) source code, object code, control code, control data, or a collection of these items (2) an aggregation of software, such as a computer program or database, that satisfies an end use function and is designated for specification, qualification testing, interfacing, configuration management, or other purposes (3) identifiable part of a software product. (ISO/IEC 24765:2009)

software release management: management of the activities surrounding the release of one or more versions of software to one or more customers, including identifying, packaging, and delivering the elements of a product. (ISO/IEC 24765:2009)

software repository: a software library providing permanent, archival storage for software and related documentation. (IEEE Std 24765)

software version ID: an explicit and immutable version identifier (name or number) inserted into each configuration item, including each individual release, that can be used to identify the exact version of the configuration item in any instance or repository.

system: combination of interacting elements organized to achieve one or more stated purposes. (ISO/IEC/IEEE 15288:2008, 4.31) (ISO/IEC TR90005:2008, 2.1) (ISO/IEC 15939:2007, 3.39)

version: (1) an initial release or re-release of a computer software configuration item, associated with a complete compilation or recompilation of the computer software configuration item (2) an initial release or complete re-release of a document, as opposed to a revision resulting from issuing change pages to a previous release. (ISO/IEC 24765:2009)

versioning: the assignment of either unique version names or unique version numbers to unique states of software configuration items, usually for a specific purpose, such as a release of the software product to an external group or the identification of a specific baseline.

2.2 Acronyms and abbreviations

The following acronyms appear within the text of this standard:

CCB Change Control Board, Configuration Change Board

CI Configuration Item

CM Configuration Management

CMDB Configuration Management Database

CMP Configuration Management Plan

CMS Configuration Management System

COTS Commercial Off-The-Shelf

CD (see CD-ROM)

CD-ROM Compact Disc, Read-Only-Memory

DVD Digital Versatile Disc

FCA Functional Configuration Audit

ID Identification

PCA Physical Configuration Audit

QA Quality Assurance

RFC Request For Change

3. Tailoring

This standard covers configuration items to be handled in any portion of the full system and software life cycle. The Supplier Configuration Item Control process may be omitted if there are no suppliers providing configuration items.

4. Audience

The primary users of this standard are assumed to be those with authority and responsibility for planning, managing, and performing CM. The user of this standard is expected to expand and supplement the minimum requirements as necessary for the development environment, specific industry, organization, and project. In addition, the audience for this standard includes all of those who contribute configuration items or who retrieve configuration items in order to use or change them.

In considering adoption of this standard, regulatory bodies should be aware that specific application of this standard may already be covered by one or more IEEE standards documents relating to quality assurance, definitions, or other matters.

Further, it is not the purpose of this standard to supersede, revise, or amend other existing standards directed to specific industries or applications.

5. The configuration management process

CM is central to, and provides essential services to, all the major processes of systems and software engineering as shown in Figure 1. All of these, along with the higher level descriptions of CM and SCM processes, are described in ISO/IEC/IEEE 12207:2008 and ISO/IEC/IEEE 15288:2008. The normative sections of this standard are mapped to these standards in Annex L.

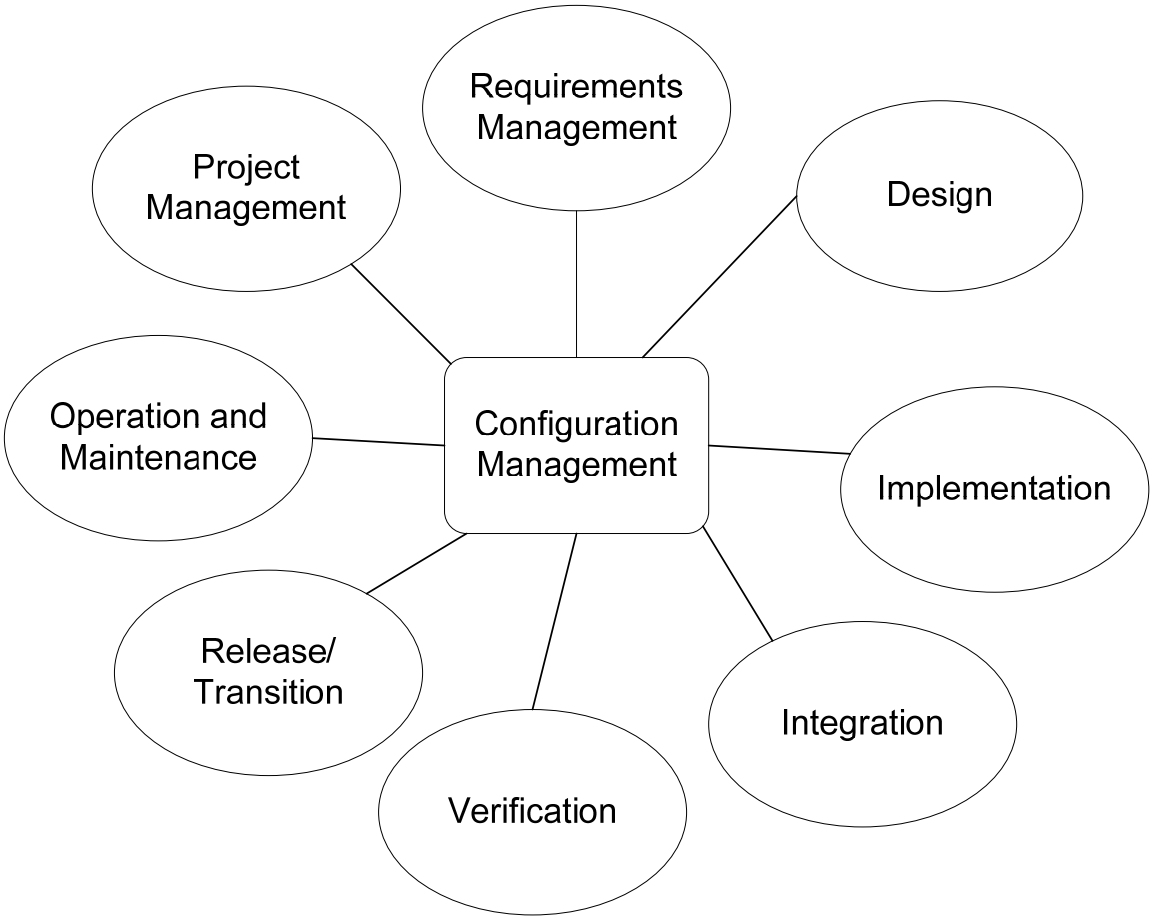

Figure 1—Example life cycle processes that CM supports

Configuration Management is essential throughout the product's life cycle, from inception (identification of the need for the product) through its end-of-life disposal. The intensity of CM activity, however, varies according to different technical processes being invoked during the life cycle.

The Configuration Management process in systems and software engineering comprises seven primary lower-level processes and two special instances of applying those lower-level processes (as shown in Figure 2), all designed to manage the Configuration Items (CIs) within a given project or for a given product, beyond the life of a single project. The primary lower-level processes are Planning, Management, Configuration Identification, Configuration Change Control, Configuration Status Accounting, Configuration Auditing, and Configuration Release Management. The special instances are Interface Control and Supplier Configuration Item Control. The items to be managed by these special instances require exercising all the primary processes. The special instances are called out separately here to highlight their importance.

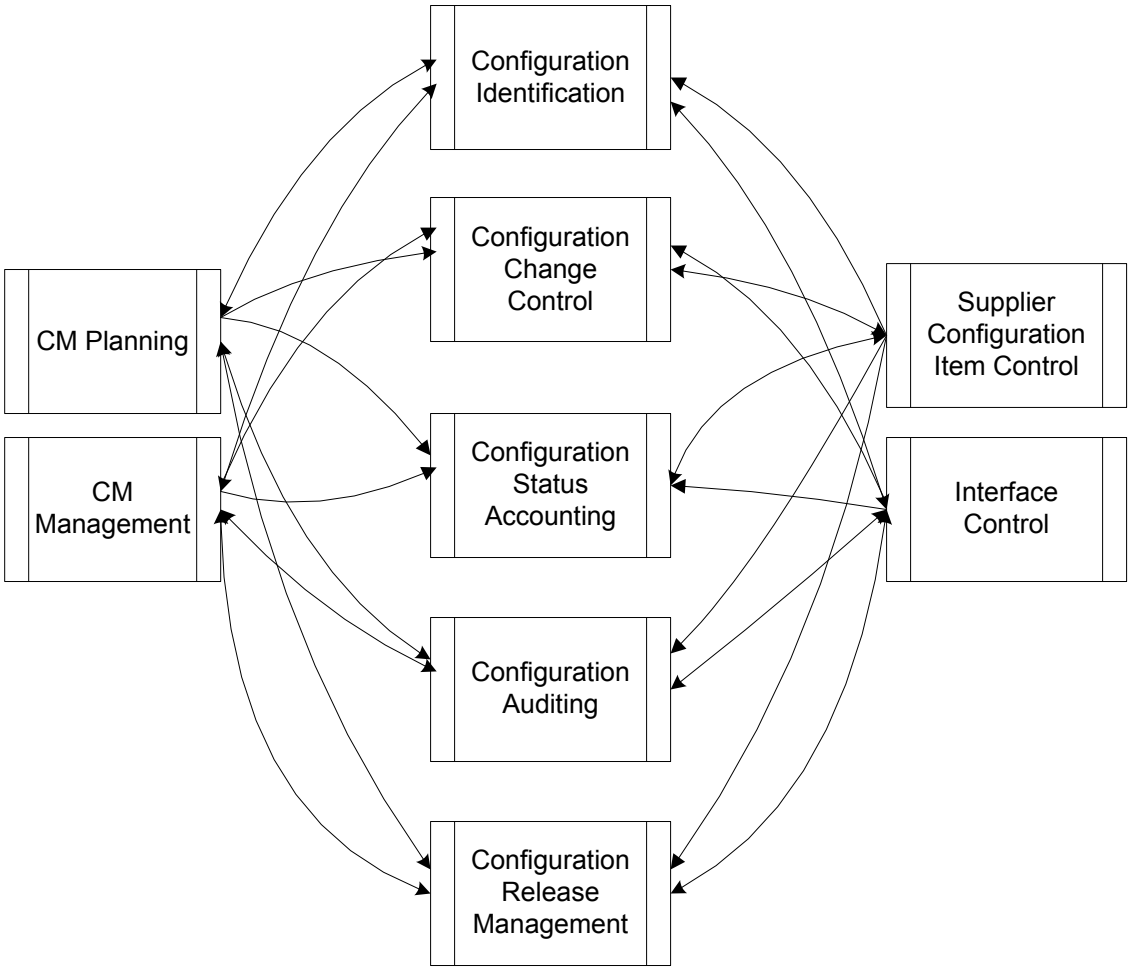

Figure 2—Lower-level processes

Additional information on how these lower-level processes are applied throughout the lifecycle is given in Annex A and following.

6. CM planning lower-level process

6.1 Purpose

The purpose of CM planning is to produce and communicate effective and workable CM plans (CMPs), whether for a project or for on-going CM services to an organization. Activities of this lower-level process determine the scope of the CM management and technical activities; identify process outputs, project tasks and deliverables; establish schedules for project task conduct, including achievement criteria, and required resources to accomplish CM project tasks.

The purpose of CM planning is to identify the activities and tasks (both technical and managerial) required to manage the configuration of the product, as specified in the requirements, be it at inception or at any other point in the product's life cycle, including eventual disposal (retirement or replacement) of the product.

NOTE—CM management information describes the allocation of responsibilities and authorities for CM activities and their management to organizations and individuals within the project structure, while CM technical information describes the technical CM activities that should be carried out, as well as the environment and tools that should be used.

6.2 Activities and tasks

See Annex D for more information on the development of the CMP.

6.2.1 Develop CM plan

6.2.1.1 Identify management information needs

The CM authority shall interview the project or management team's needs for information about the status of configuration items (CIs).

6.2.1.1.1 Determine reporting needs

The types (e.g., number of defect reports associated with configuration items) of information that will be required shall be determined.

6.2.1.1.2 Determine reporting frequency

The frequency with which required information shall be reported shall be determined.

6.2.1.2 Identify information needed to manage CM activities

The CM authority shall consider the following topics for inclusion in the CMP. Any omitted topics shall be listed with the reasons they are omitted.

a) the plan's purpose, scope of application, key terms, and references

b) the responsibilities and authorities for managing and accomplishing the planned CM activities, including those for managing third-party configuration items

c) CM activities and tasks to be performed

d) the required coordination of CM activities with the other activities in the project or organization, and the interfaces necessary to achieve coordination

e) tools and physical and human resources required for execution of the plan, including those needed for environments, infrastructure, tools, techniques, processes, equipment, and training

f) tools to be used in the performance of CM activities

g) how the plan will be maintained while in effect

h) sizing and estimation of effort and duration for each of the CM activities

i) milestones such that the activities are broken into manageable elements of work

j) estimated costs of the activities (human and material resource costs)

k) resourcing of planned CM activities (time allocations as well as human and material resources)

l) dependencies between CM activities and other project activities

m) forward and backward task dependencies

n) naming scheme for product builds

NOTE—See Annex K for specific information about software builds.

o) performance goals and required service levels to be achieved

p) configuration audits to be performed (scope, duration, outputs)

q) risks to the successful execution of the CMP and appropriate mitigation strategies and levels of effort required to mitigate; communication of residual risks to the appropriate persons

NOTE—A detailed description of the contents of the CMP is provided in Annex D.

6.2.1.3 Document CM plan

A CMP shall be established either in stand-alone form or by reference to other locations such as other documents or automated systems. Location information shall be provided to those responsible for performing CM activities and to their management.

NOTE—As with any other activities in a technical project, the CM activities should be allocated among the project team members. These team members may be part of a development group, test group, or other. CM planning includes negotiating for agreement to allocate resources to CM activities.

7. CM management lower-level process

7.1 Purpose

The purpose of CM management is to implement, monitor, control, and improve CM services. This includes determining the status of the CM activities to ensure that the CM activities and tasks are carried out according to plans and schedules, within projected budgets, and satisfy technical objectives. In addition, known risks are monitored and the effectiveness of mitigation strategies assessed. Project metrics are collected and analyzed for leading indication of CM issues to be managed. This lower-level process includes redirecting the CM activities as appropriate to correct identified deviation and variations from other CM project management or technical processes. Redirection may include re-planning as appropriate.

NOTE—ISO/IEC/IEEE 16085, ISO/IEC/IEEE 15939, ISO/IEC/IEEE 16326, and IEEE Std 1490™ should be consulted for further information.

7.2 Activities and tasks

7.2.1 Manage implementation of CMP

Human and physical resources needed for environments, infrastructure, tools, techniques, processes, equipment, and training shall be procured.

All personnel resourced for CM tasks shall be informed of their responsibilities as documented in the CMP and trained to carry them out.

Tools and environments shall be properly installed and configured.

7.2.2 Monitor CM activities

7.2.2.1 Monitor resource usage

If there is any shortfall in planned resources, then any resultant added risks to the CMP shall be communicated to the appropriate persons for development of mitigation actions. The CMP shall be modified as necessary upon agreement of appropriate management on appropriate mitigation actions. Should any residual risk remain, the CM authority and the Project Manager shall decide on an appropriate course of action.

7.2.2.2 Monitor progress

CM activities shall be tracked against the CM schedule as a regular part of Project Management. This includes planned deliveries and deliverables from third parties.

7.2.2.3 Monitor risks

Monitor the effectiveness of risk management plans and the emergence of new risks.

7.2.2.4 Identify variances

Adherence to the plan shall be tracked such that variances are identified and rectification actions taken.

7.2.2.5 Update plans

The CMP shall be updated as required when information that affects the currency of the CMP changes.

The CMP shall be updated as necessary to ensure continued CM planning during the life cycle of the product.

At a minimum, periodic reviews of this plan shall occur at the start of each iteration/phase of the emerging product.

At that time, proposed changes, if any, shall be evaluated and, with the requisite approval, implemented within the plan.

All CMP changes shall be communicated to the project team.

NOTE—For a discussion of configuration management planning in the system or software life cycle, see Annex F.

8. Configuration identification lower-level process

8.1 Purpose

The purpose of configuration identification is to determine naming schemes for configuration items (CIs), identify the items that require control as CIs, and to apply appropriate names to them. Additionally, the physical and functional characteristics of the CIs are identified.

The scope of configuration identification includes:

a) determining the CIs that are to be managed and determining what documentation information is to be used for describing the physical and functional characteristics of each CI

b) planning for the collection, storage, retrieval, and change control of baselined versions of the items and their descriptive documentation information

c) establishing and maintaining associations between versions of each item and its descriptive information

d) establishing versioned assemblies or collections of CI versions that satisfy the totality of end use functions

e) establishing and maintaining associations between versions of such assemblies or collections and the descriptive documentation information of the physical and functional characteristics of the assembly or collection, and

f) describing the product structure through the selection of CIs and identification of their inter-relationships.

8.2 Activities and tasks

8.2.1 Establish the structure and hierarchy of configuration items

A structure and hierarchy shall be determined based on the product's architecture (that is, its components and the relationship(s) among them). Software, firmware, hardware, and documentation/description items shall all be taken into account. A branching model facilitating traceability of changes to the software items shall be established in the development and release environments.

8.2.2 Identify configuration items

Using established selection criteria, CM personnel shall work with the producers of project artifacts to identify which items are to be controlled and the descriptive documentation to be associated with each. These items shall be identified as CIs.

Product configuration information includes both product definition and product operational information.

NOTE 1—CM personnel should work with the producers of project artifacts to verify this list is complete.

NOTE 2—Controlled items may be intermediate and final outputs. For example, approved requirements documents become CIs at the point when changes to them must go through a defined change control process. The approved requirements are then baselined as CIs. That is, that CI becomes the baseline on which design activities are based.

NOTE 3—Items that are likely to need controlling include, but are not limited to, baselined requirements specifications, interface specifications, designs, code, builds, build data, database-related items such as triggers, schema and SQL scripts, unit and coverage tests, and the standards that were used to create such items. In addition, the following are typically included: design drawings, parts lists, reference models, baselined models or prototypes, and maintenance and operating manuals. The determining factor is whether an item or information will be needed if the project needs to reinstate a previous baselined position in the life cycle to once again move forward on the build cycle.

8.2.3 Describe configuration items

Once an item has been identified as a CI, the producer of the item shall produce an item description and maintenance information.

8.2.4 Name configuration items

8.2.4.1 Establish naming convention

CM shall ensure a consistent naming convention is designed to enable each CI to be uniquely named. CM shall ensure that the naming convention takes into account the need to accommodate multiple versions of items that may arise from the creation of a number of different baselines.

CM shall ensure that the naming scheme is extensible and capable of accommodating configuration items originating from third party suppliers.

NOTE 1—This activity includes such items as reference implementations, hardware prototypes, and COTS, subcontracted, and supporting software.

NOTE 2—The design of the naming system should consider efficient storage, retrieval, tracking, reproduction, and distribution of CIs.

8.2.5 Assure CIs are placed in CM repository

8.2.5.1 Establish controlled repositories

CM shall identify the controlled repositories for the CIs and describe how the identified CIs are to be controlled in the appropriate repository.

NOTE—Additional information regarding software asset management can be found in the ISO/IEC 19770 series of standards.

For each electronic repository, the branching structure, format, location, documentation requirements, receiving and inspection requirements, and access control procedures shall be specified.

8.2.5.2 Acquire electronic CIs

After electronic CIs are identified, named, and completed, they shall be deposited in the appropriate CM repository (e.g., library).

8.2.5.3 Acquire physical CIs

CM shall ensure that any physical configuration items that are developed or acquired are appropriately labeled and stored.

8.2.5.4 Establish criteria for baselines

Criteria for establishing a CI as a baseline item (an approved configuration item) shall be defined and incorporated in the CMP. The criteria shall define the formal approval(s) required to assign baseline status to a CI, such as manager sign-off for a project plan, or quality criteria for a software or system product.

NOTE—A baseline provides a logical basis for comparison. A specific version of a single work product by itself, or a set of work products together can be established as a baseline. During the course of product development, a series of baselines is established, enabling assessment of the evolving product's maturity at different points in time.

8.2.5.5 Define how baselines are established

This task shall include identifying:

a) The events that establish a baseline

b) The items that are to be controlled in the baseline

c) The procedures used to establish and change the baseline

d) The authority required to approve changes to the approved baselined items

8.2.5.6 Identify baselines

During each iteration/phase of a development project, newly developed CIs and new versions of pre-existing CIs shall be identified as CIs.

At the close of each iteration/phase, approved CIs shall be baselined for the project.

CM shall ensure that all updates, deletions, and additions to identified items are performed only as an outcome of the change control process.

8.2.5.7 Establish change control process

CM activities shall ensure that a documented change control lower-level process describes how changes to baselines and their associated items are requested and dispositioned.

CM activities shall ensure that the appropriate approvals bodies are in place to make decisions about requested changes.

CM activities shall ensure that data is included with any change to provide traceability to the original Request for Change.

8.2.5.8 Establish physical storage procedures

CM shall specify procedures for the physical storage of documents needed for disaster recovery, such as procedures for access to stored records and documents and stored physical media, reference hardware, and electronic media, including the physical marking and labeling of items.

NOTE—Off-site archiving, data retention periods, and technical data recovery procedures should also be described.

8.2.5.9 Establish CM procedures

CM procedures shall describe how to retrieve and reproduce controlled items from the repository.

These activities shall include verification of marking and labeling, tracking of controlled copies, and protection of proprietary and security information.

8.2.5.10 Assure system maintenance for CM repository reliability

CM shall ensure that baselines are reproducible from the same sources from which they were originally created.

CM shall ensure that backup storage of its baselined data is maintained to facilitate technical data recovery.

CM shall ensure that a technical data recovery procedure is defined and deployed.

For a discussion on applying configuration identification in the software life cycle, see Annex G.

9. Configuration change control lower-level process

9.1 Purpose

The purpose of configuration change control is to maintain the integrity of the product in all of its states, from requirements to a validated working product, as changes to it are needed both under development and post-release.

Configuration items may be constituent or composite. A baselined software or system product is a composite configuration item: It is composed of more than one constituent item.

Configuration change control for constituents differs from change control for product or system baselines. Each may be managed by different processes and tool sets and with different levels of formality.

The change control lower-level process applies to CIs for which a baseline has been established, that is, for CIs which have been approved and placed into the appropriate repository. It does not apply to items which have not yet been submitted to a CM repository for the first time and are still under development.

Examples of how configuration control can be implemented in a project are in Annex H.

9.2 Activities and Tasks

9.2.1 Establish change control infrastructure

9.2.1.1 Designate items subject to change control

The following types of items are the minimum that shall be subject to change control:

a) Constituent CIs as they participate in larger components (e.g., with other CIs in software builds)

b) Each configuration baseline as a separately identified configuration item

c) Specification of the environment and tool chains used in producing the baseline

NOTE—Items that are subject to control include, but are not limited to, baselined requirements specifications, interface specifications, designs, code, builds, build data, database-related items such as triggers, schema and SQL scripts, unit and coverage tests, and the standards that were used to create such items. The determining factor is whether an item or information will be needed if the project needs to reinstate a previous baselined position in the life cycle to once again move forward on the build cycle.

9.2.2 Establish change evaluation criteria and authorities (CCB)

To ensure consistency in the evaluation and disposition of change requests, the following infrastructure elements shall be established:

a) Those who shall have the authority to approve or reject changes shall be established for each product and project baseline to assure technical and managerial evaluation of proposed changes, with the authority and empowerment to approve/reject the changes.

b) The authorities shall include representatives from all stakeholder organizations (each functional area that may be affected by changes in the baseline).

c) The change authorities shall used pre-established criteria when evaluating change requests.

NOTE—The sets of criteria used may differ, depending on such factors as constraints of the project schedule, resources available, stability of the product or its components, etc.

9.2.3 Establish change request form

To facilitate the formal change management process, a change request form shall be used. This form shall be incrementally completed as it passes throughout the entire change life cycle, from origin of the request to its final disposition. A change request form shall be established with at least the following standard fields:

a) Description of proposed change and rationale/purpose

b) State of the change (e.g., open, approved/rejected, implemented, tested)

c) Affected baseline

d) Outcome of analysis of impact on the project

e) Resolution

f) Approvals

9.2.4 Control changes to all configuration items

9.2.4.1 Control changes to constituent configuration items

Each change to a constituent configuration item shall be controlled as the product evolves from one baseline to the next.

For controlling constituent configuration items, the following attributes shall be available in the CMDB for each change request:

a) **Originator**: The originator of the change request shall be recorded.

b) **Impact analysis**: When committing the change to the CMDB, concurrence conflicts shall be evaluated (may be done by the CMDB automatically).

c) **Rationale and approval**: The rationale (provided, e.g., by reference to a CR number or defect report number) shall be recorded by the CM system as the commit/check-in description.

d) **Notification**: The primary user of the item shall be notified about the change. Records of informative notification shall be available for status accounting reports and in other forms of on-demand communication (e.g., subscribing to change notifications of a configuration item).

e) **Reversibility**: The CM system shall provide facilities to revert any given change to a configuration item.

f) **Set of Changes**: The list of artifacts affected by the change shall be recorded in the CM system.

g) **Audit trail**: Persistence of all the records shall be implemented.

9.2.4.2 Control changes to baselines

Each change to a baseline shall be controlled using the formal lower-level change control process defined in the CMP.

For controlling baselined software or systems as CIs, the following attributes shall be available in the CMDB:

a) **Attribution**: Attribution shall be provided to the CCB that is accountable for the affected baseline.

b) **Impact analysis**: A documented impact analysis reviewed by all the CCB members shall be recorded. Impact analysis shall consider severity of any problem to be corrected by the change as well as how frequently or how widely such problem will affect the product's users. Impacts to the product development project shall also be considered.

c) **Rationale and approval**: The CCB shall approve/reject the change using its established decision process and shall provide the reason for their decision.

d) **Notification**: Notification to affected parties shall be part of the formal process.

e) **Reversibility**: Baseline change control shall be supported by versioning systems that allow individual reversibility of constituent configuration items. This, aided by the set of changes requirement, assures full reversibility.

f) **Set of Changes**: The affected baseline and the impacted artifacts (from the impact analysis) form the set of changes to apply to the baseline and shall be documented.

g) **Audit trail**: A record of the change request forms containing the process evidence shall be kept for auditability purposes.

9.2.5 Verify approved disposition of change requests

9.2.5.1 Verify implementation of approved changes

Implementation, testing, and traceability of each approved change shall be verified and documented for traceability. Unsuccessful changes shall be backed out or corrected before a new baseline is established.

9.2.5.2 Verify non-implementation of deferred/rejected items

For those change requests which have been denied, their disposition shall be recorded and verified.

9.2.5.3 Report disposition of change requests

Change requests and their disposition shall be tracked and the information made available to management and project personnel as appropriate.

For a discussion on applying configuration control in a project, see Annex I.

10. Configuration status accounting lower-level process[4]

10.1 Purpose

The purpose of CM is to keep track of all assets that have been designated as CIs. The purpose of configuration status accounting records, retrieves, and reports critical information about assets under configuration control to management and the project team.

Information about the status of assets can aid in:

a) Ascertaining the results of project work during a given period and develop estimates-to-complete at any point in the project. Example: The number of requirements, when compared to the number built, can indicate progress to date.

b) Ascertaining the developing product's status with regard to stability and functional completion. Example: The number of changes implemented and pending against a feature or component can indicate either stability or volatility.

c) Verifying control over assets

d) Satisfying external compliance audit requirements (e.g., SAS70, SOX, etc.) as needed

NOTE 1—Examples of external compliance audit requirements include, but are not limited to, SAS70 and SOX audits.

NOTE 2—"Status accounting" does **not** refer to the project management function of tracking of CM activities and tasks themselves (see 7.2.2).

10.2 Activities and tasks

10.2.1 Verify CI status information needs

The CMP shall be reviewed to verify:

— The types of information that project members will expect

— The frequency with which information is to be reported or whether any report types will be required on demand.

At a minimum, data elements to meet information needs shall include:

a) The status (such as being changed, stable, archived) of each of the current configuration items

b) Identification of the current baseline configuration of all items comprising the product

c) The state of all changes from change request through change disposition, whether implemented, rejected, deferred, or pending

d) Relationship data that enables traceability from top to bottom (requirements specification to tests to implementation) and bottom to top for all the configuration items

NOTE—In some cases, reports include additional information about changes, such as whether they constitute deviations or waivers (e.g., pertaining to contractual requirements).

[4] The term "accounting" has traditionally been used in naming this lower-level CM process, with the sense of "an account of happenings," a narrative, or record of events.

10.2.2 Verify CM mechanisms are properly set up to support information needs

CM shall verify that the chosen CM tools/mechanisms enable capture of the defined data elements in the repository and that the repository can be searched to provide the needed data for reports.

10.2.3 Account for all CIs

Each configuration item (CI) that has been identified and placed under configuration control shall be accounted for at each stage of its life, from its initial identification as a CI through its end-of-life. That is, each CI shall be findable in its designated location, with the following attributes:

a) Its origin (date and source)

b) Its location

c) Its approved version(s)

d) Identity and status of requested changes to it

e) The implementation status and efficacy of approved changes

f) The CI's parent and successor CIs (for traceability)

10.2.4 Report on status of CIs

CM shall use the information needs identified in the CMP as a basis for designing and preparing reports on the status of CIs, the status of all requested changes, and other such information as designated by project personnel and organizational management in the activity. Status accounting reports shall be made available to the approriate management and staff members.

10.2.5 Report discrepancies from audits

Any CM discrepancies and deficiencies discovered during audits shall be recorded and reported to project and organizational management. Actions taken to correct these, as well as results of those actions, shall also be recorded and reported.

For a discussion on applying configuration status accounting in a development effort, see Annex J.

11. CM configuration auditing lower-level process

11.1 Purpose

The purpose of configuration auditing is to objectively assess the integrity of the product both from a functional perspective (how the different technical processes of product development—from requirements to testing—were performed) and from a physical perspective (how the as-is product was built and changes were applied).

11.2 Activities and Tasks

11.2.1 Perform functional configuration audits

Functional configuration audits shall be performed during the product development life cycle to assess the proper transition of functions from requirements to the built product at least once before releasing the product to a production environment.

Functional configuration audits shall be performed by a role that assures the objectivity of the audit.

NOTE 1—This audit type is based on the evolution of the different forms of the product, from its inception (in the form of user requirements) up to the actual working product under test.

NOTE 2—Functional audits may be executed as soon as the product evolves to a newer form or at the end of the development, or they can be deferred to be executed prior to the release of the product.

11.2.1.1 Inspect traceability

a) Inspect traceability between requirements and the rest of the product models (use case, design, deployment, implementation, etc.)

b) Inspect traceability between requirements and the test artifacts that express that the product requirements have been implemented

c) Inspect traceability between the tests case and its execution

NOTE—It is not the function of CM Functional Audits to evaluate conformance to requirements (rather, testing is) but to assure that completeness was achieved in the transition from requirements to the built product (all requirements are implemented and tested).

11.2.2 Perform physical configuration audits

Physical configuration audits shall be performed during product development to assess the (right) product is being properly assembled and changes are managed on the different product artifacts at least once before releasing the product to the production environment.

NOTE—Physical audits may be executed as soon as the product evolves to a newer form or at the end of the development, or they can be deferred to be executed prior to the release of the product.

Physical configuration audits shall be performed by a role that assures the objectivity of the audit.

11.2.2.1 Inspect physical attributes

a) Identify work products, including those that are both part of the end product or the original models or collateral artifacts

b) Inspect that proper build attributes are present

NOTE—For software, the build manifest lists elements consisting of a set of attributes and sections.

c) Inspect that proper change management elements are present on each change performed to the product configuration (records, audit trails, rationale, impact analysis, reversibility, attribution)

NOTE—This is usually thoroughly audited for baseline change control rather than for CI change control.

d) Confirm that the product characteristics and actual differences from the specified requirements are present in the release notes (list of known defects, testing limitations, product version identification, etc.)

11.2.3 Perform baseline configuration audit

Baseline configuration audits shall be performed during the product cycle to assess the proper creation of baselines.

Baseline configuration audits shall be performed by a role that assures the objectivity of the audit.

11.2.3.1 Inspect traceability

a) Baseline CI readiness (allowing only CI in the proper state to be part of the baseline)

b) Baseline identification

c) Baseline approval

For a discussion on applying configuration audits in software projects, see Annex K.

11.2.4 Record and report nonconformities

In all of the audits described above, CM auditors shall determine if any nonconformities exist, and, if so, record them with sufficient information to later verify their correction.

Any discrepancies detected during the above audits shall be reported to the appropriate persons, as defined in the CMP, for correction.

Actions taken to correct these, as well as results of those actions, shall also be recorded and reported.

11.2.5 Verify discrepancy resolution

For all audits described above, upon notification that the discrepancies have been resolved, problem areas shall be re-audited and the results reported, as stated above.

12. Interface control lower-level process

12.1 Purpose

The purpose of the interface control lower-level process is to manage the potential interfacing effects that hardware, system software, and support software, as well as other projects and deliverables, have on the project. Interface control activities coordinate changes to the project CIs with changes to interfacing items.

NOTE—Interfaces represent “agreements” between different development efforts. Each party is constrained by the requirements of the interface. Thus, each interface represents at least three CIs: the interface specification itself, and components on either side of the interface.

An interface may be between components developed internally to the project or between project components and components developed externally.

The interface description CI describes the interface entity characteristics of one or more systems, subsystems, domains, hardware items, software items, manual operations (processes) or other system components. It presents interface characteristics, including systems or configuration items performing the interface (including human-system and human-human interfaces), standards and protocols, responsible parties, interface operational schedule, and error handling. It includes interface diagrams to depict the interfaces. It defines existing or permanent interface characteristics and those that are being developed or modified.

12.2 Activities and Tasks

12.2.1 Identify the product's key interfaces

For each interface, the following shall be defined:

a) The nature of the interface (data, hardware, software)

b) The affected organizations

c) Technical specifications

12.2.2 Control the interface specifications

Specifications for each interface between internally developed components and between the project elements and external elements shall be placed under CM control in the designated repository.

Specifications shall be subject to the project's CM control, audit, and accounting processes. For any CCB established specifically to control interfaces, its responsibilities and procedures shall be defined.

13. Supplier configuration item control lower-level process

13.1 Purpose

The purpose of the supplier configuration item control lower-level process is to manage the incorporation of items developed outside the project environment (e.g., by third parties) into the project CIs in order to support added accountabilities for organizational and legal relationships. Acquired CIs can come from any supplier — a vendor, a customer, another project, or other source. Included are CIs developed by contract whether acquired in finished form or as individual elements.

13.2 Activities and Tasks

13.2.1 Include handling of acquired items in the CMP

The CMP shall describe

— how acquired items will be received, tested, and placed under CM

— how changes to the supplier's items are to be processed

— whether and how the supplier will participate in the project's change management

The activities to incorporate the acquired items into the project CIs and to coordinate changes to these items with their development organizations shall be defined in the CMP.

For subcontracted items, the following shall be described:

a) What CM requirements, including a CMP, are to be part of the supplier's agreement

b) How the supplier will be monitored for compliance

c) What configuration audits and reviews of supplier items will be held

d) How external items will be tested, verified, accepted, and merged with the project CIs

e) How proprietary items will be handled for security of information and traceability of ownership (e.g., copyright and royalties)

f) How changes are to be processed, including any supplier participation

13.2.2 Place acquired items under CM

In addition to the activities named above, the acquired items shall be appropriately marked and placed in the designated project and/or product repository and controlled, audited, and accounted for as described in the CMP.

14. Release management lower-level process

14.1 Purpose

The purpose of the release management lower-level process is to assure that the proper set of deliverables (including documentation and ancillary materials) is delivered to the designated receiving party in the designated form to the designated location.

A release is a version of software or a system under CM that is made formally available to a wider community. This includes both external releases to customers and internal releases, for example, to another internal development group or to a testing group.

14.2 Activities and tasks

14.2.1 Delineate general requirements

Product releases shall conform to a release policy, follow a release plan, contain the specified release contents, and be in the specified release format. They shall be delivered in the appropriate manner and tracked as they progress from state to state. When a release has reached its end-of-life, it shall be archived, made unavailable via normal channels, and marked as such in the tracking mechanism.

Master copies of all configuration items in a release and the release itself shall be maintained for the life of the product.

14.2.2 Define release policy

Release management personnel shall provide assistance to the project managers and developers to:

a) Define the delivery qualification criteria for delivering a release

b) Define the person(s) or group(s) who will make the decision to deliver a release

14.2.3 Define release planning

Release management personnel shall provide assistance to the management and development teams to:

a) Define the types of releases (e.g. to other development teams, to Quality Assurance (QA), to beta customers, for General Availability (GA) to customers, to an escrow account)

b) Define to whom releases will be given and when (or under what circumstances)

14.2.4 Define release contents

Release management personnel shall provide assistance to the management and development teams to define what components must be included in the release:

a) The results of a build (or possibly builds from multiple components)

b) Supporting components (e.g., release notes, ReadMe files, Help files, operations manuals, and other documentation)

Most release components will already be configuration items. Some, like release notes or version description documentation, are configuration items created in the process of creating the release. The release itself becomes a configuration item.

14.2.5 Define release format and distribution

Release management personnel shall provide assistance to the management and development teams to:

a) Define the format of the delivered release (e.g. CD/DVD ISO files, tar files, turn-key sytem, etc.)

b) Define how the release will be distributed (e.g. on CD-ROM, copied to a pre-designated location on a file server, etc.)

14.2.6 Define release tracking

Release management personnel shall track releases throughout the product's life cycle. At a minimum, the following data shall be captured:

a) The date of the release

b) To whom the release was delivered

c) The "state" of the release

d) The release environment (O/S, patch level)

e) What release is superseded by this release

In the case of a shrink-wrapped software product, release management personnel track a release from the QA state to the "gold-master" state. In a production shop, tracking follows the release from the QA state through the Staging state and finally to the Production state (on individual servers, if the business requires it).

Release tracking is often required for compliance with external laws and regulations. It may also be a contractual requirement.

14.2.7 Deliver approved releases

Once a product build (assembly) has been approved for release, it shall be identified and baselined as an approved release and delivered to the designated receiving party in the designated form to the designated location. Release tracking information (defined above) shall be made available to all stakeholders and other interested parties.

14.2.8 Archive

When a release has reached its end-of-life, the CM authority shall:

a) Archive the release in an officially designated location

b) Make the release unavailable via normal channels (e.g., removed from the designated location on a file server)

c) Mark the release as “archived” in the release tracking data

Annex A

(informative)

CM lower-level process models

A.1 General

The following lower-level process models are closely aligned to the requirements of Clause 6 to Clause 14:

- CM planning process
- CM management process
- Configuration identification process
- Configuration change control process
- Configuration status accounting process
- Configuration auditing process
- Interface control process
- Release management process

NOTE—There is no specific lower-level process for Clause 13. These requirements are addressed by process concerns in the planning and configuration identification processes in the format indicated by the ISO/IEC guidelines for process description [B13][5].

[5] The numbers in brackets correspond to those of the bibliography in Annex M.

A.1.1 CM planning

Name	CM planning	
Purpose	The purpose of CM planning is to produce and communicate effective and workable CM plans, whether for a project or for on-going CM services to an organization.	
Outcomes	As a result of successful implementation of this process: 1. the scope of the work for the CM project is defined 2. the feasibility of achieving the goals of the CM project with available resources and constraints is evaluated 3. the tasks to be undertaken to perform the work are identified 4. the resources necessary to perform the work are identified 5. the tasks to be performed and resources necessary to perform the work are sized and estimated 6. interfaces between elements in the CM project, and with other CM projects, are identified 7. plans for the execution of the CM project are developed 8. plans for the execution of the CM project are activated	
Requirements traceability	IEEE 828 6.2.1.1 IEEE 828 7.2.1 IEEE 828 13.2 IEEE 828 14.2.3 IEEE 828 12.2.1 IEEE 828 13.2.1 IEEE 828 6.2.1.2	Identify management information needs [1] Manage implementation of CMP [4] Activities and tasks [1] Define release planning [7] Identify the product's key interfaces [6] Include handling of acquired items in the CMP [1] Identify information needed to manage CM activities [1]

A.1.2 CM management

Name	CM management	
Purpose	The purpose of CM management is to implement, monitor, control, and improve CM services.	
Outcomes	As a result of successful implementation of this process: 1. resources to perform the CM project are obtained 2. progress of the CM project is monitored and reported 3. interfaces between elements in the CM project, and with other CM projects and organizational units, are monitored 4. actions to correct deviations from the plan and to prevent recurrence of problems identified in the CM project are taken when SSCM project targets are not achieved 5. CM project objectives are achieved and recorded	
Requirements traceability	IEEE 828 7.2.2.1 IEEE 828 7.2.2.3 IEEE 828 7.2.2.5 IEEE 828 7.2.1 IEEE 828 7.2.2.2 IEEE 828 7.2.2.4 IEEE 828 14.2.6	Monitor resource usage [4] Monitor risks [2] Update plans [2] Manage implementation of CMP [1, 3] Monitor progress [4] Identify variances [4] Define release tracking [2]

A.1.3 Configuration identification

Name	Configuration identification	
Purpose	The purpose of configuration identification is to determine naming schemes for configuration items (CIs), identify the items that require control as CIs, and to apply appropriate names to them. Additionally, the physical and functional characteristics of the CIs are identified.	
Outcomes	As a result of successful implementation of this process: 1. the product configuration is defined 2. items requiring configuration are identified 3. internal and delivery baselines are established	
Requirements traceability	IEEE 828 8.2.1 IEEE 828 8.2.3 IEEE 828 8.2.5.6 IEEE 828 8.2.2 IEEE 828 8.2.5.3 IEEE 828 14.2.4	Establish the structure and hierarchy of configuration items [1] Describe configuration items [2] Identify baselines [2, 3] Identify configuration items [1, 2] Acquire physical CIs [2] Define release contents [2]

A.1.4 Configuration change control

Name	Configuration change control	
Purpose	The purpose of configuration change control is to maintain the integrity of the product in all of its states, from requirements to a validated working product, as changes to it are needed both under development and post-release.	
Outcomes	As a result of successful implementation of this process: 1. CI change requests are recorded and classified 2. CI change requests are assessed using defined criteria 3. approved changes to CIs are implemented 4. CI changes are verified 5. unsuccessful CI changes are reversed or remedied	
Requirements traceability	IEEE 828 8.2.5.6 IEEE 828 9.2.5.1 IEEE 828 9.2.4.1 IEEE 828 9.2.5.2	Identify baselines [3] Verify implementation of approved changes [4, 5] Control changes to constituent configuration items [1] Verify non-implementation of deferred/rejected items [2]

A.1.5 Configuration status accounting

Name	Configuration status accounting	
Purpose	The purpose of CM is to keep track of all assets that have been designated as CIs. The purpose of configuration status accounting is to record, retrieve, and report critical information about assets under configuration control to management and the project team.	
Outcomes	As a result of successful implementation of this process: 1. CM reporting needs are identified 2. CM reports are prepared according to defined criteria 3. CM reports are made available to affected parties	
Requirements traceability	IEEE 828 6.2.1.1.1 IEEE 828 10.2.1 IEEE 828 10.2.4 IEEE 828 6.2.1.1.2 IEEE 828 10.2.3 IEEE 828 10.2.5	Determine reporting needs [1] Verify CI status information needs [1] Report on status of CIs [2, 3] Determine reporting frequency [1] Account for all CIs [1] Report discrepancies from audits [2]

A.1.6 CM configuration auditing

Name	CM configuration auditing	
Purpose	The purpose of configuration auditing is to objectively assess the integrity of the product both from a functional perspective (how the different technical processes of product development—from requirements to testing—were performed) and from a physical perspective (how the as-is product was built and changes were applied).	
Outcomes	As a result of successful implementation of this process: 1. the scope and purpose of each CM audit is defined 2. the objectivity and impartiality of the conduct of CM audits and selection of auditors are assured 3. conformity of functional, physical, baseline, and release configurations to requirements is determined 4. nonconformities are recorded 5. nonconformities are communicated to those responsible for corrective action and resolution 6. corrective actions for nonconformities are verified	
Requirements traceability	IEEE 828 10.2.5 IEEE 828 11.2.1.1 IEEE 828 11.2.2.1 IEEE 828 11.2.3.1 IEEE 828 11.2.5 IEEE 828 11.2.1 IEEE 828 11.2.2 IEEE 828 11.2.3 IEEE 828 11.2.4	Report discrepancies from audits [4] Inspect traceability [3] Inspect physical attributes [3] Inspect traceability [3] Verify discrepancy resolution [6] Perform functional configuration audits [1, 2] Perform physical configuration audits [1, 2] Perform baseline configuration audit [1, 2] Record nonconformities [4, 5, 6]

A.1.7 Interface control

Name	Interface control	
Purpose	The purpose of the interface control lower-level process is to manage the potential interfacing effects that hardware, system software, and support software, as well as other projects and deliverables, have on the project. Interface control activities coordinate changes to the project CIs with changes to interfacing items.	
Outcomes	As a result of successful implementation of this process: 1. CM interfaces are defined 2. CM interfaces are monitored	
Requirements traceability	IEEE 828 12.2.2 IEEE 828 12.2.1	Control the interface specifications [1, 2] Identify the product's key interfaces [1]

A.1.8 Release management

Name	Release management	
Purpose	The purpose of the release management lower-level process is to assure that the proper set of deliverables (including documentation and ancillary materials) is delivered to the designated receiving party in the designated form to the designated location.	
Outcomes	As a result of successful implementation of this process: 1. the release content is defined 2. release format requirements are identified 3. releases are assembled 4. releases are delivered 5. end-of-life releases are archived 6. release information is communicated to interested parties	
Requirements traceability	IEEE 828 14.2.4 IEEE 828 14.2.7 IEEE 828 14.2.1 IEEE 828 14.2.5 IEEE 828 14.2.8	Define release contents [1] Deliver approved releases [4, 6] Delineate general requirements [3, 5] Define release format and distribution [2] Archive [5]

A.2 Related processes

The following processes are implied in the requirements associated with Clause 6 to Clause 14:

- CM repository management
- Process definition
- Information item management

These lower-level process models should be considered as informative.

A.2.1 CM repository management

Name	CM repository management	
Purpose	The purpose of this process is to establish the CM repository, acquire the configuration items, and maintain the repository.	
Outcomes	As a result of successful implementation of this process: 1. the CM repository requirements are identified 2. the CM repository is established 3. the CM repository items are obtained 4. the CM repository is maintained 5. the CM repository is controlled	
Requirements traceability	IEEE 828 7.2.1	Manage implementation of CMP [2]
	IEEE 828 8.2.5.1	Establish controlled repositories [1]
	IEEE 828 8.2.5.10	Assure system maintenance for CM repository reliability [5]
	IEEE 828 13.2.2	Place acquired items under CM [3]
	IEEE 828 8.2.5.2	Acquire electronic CIs [3]
	IEEE 828 10.2.2	Verify CM mechanisms are properly set up to support information needs [2]
	IEEE 828 7.2.2.1	Monitor resource usage [5]

A.2.2 Process definition

Name	Process definition	
Purpose	The purpose of the process definition process is to enable the effective management and implementation of all CM processes.	
Outcomes	As a result of successful implementation of this process: 1. processes are established 2. roles and responsibilities needed to support processes are defined 3. the effectiveness and efficiency of processes are continually improved in line with SSCM management objectives	
Requirements traceability	IEEE 828 9.2.2	Identify configuration items [2]
	IEEE 828 8.2.1	Establish the structure and hierarchy of configuration items [1]
	IEEE 828 8.2.4.1	Establish naming convention [1]
	IEEE 828 8.2.5.5	Define how baselines are established [1]
	IEEE 828 8.2.5.8	Establish physical storage procedures [1]
	IEEE 828 8.2.5.10	Assure system maintenance for CM repository reliability [1]
	IEEE 828 9.2.2	Establish change evaluation criteria and authorities (CCB) [1, 2]
	IEEE 828 9.2.4.1	Control changes to constituent configuration items [1]
	IEEE 828 12.2.2	Control the interface specifications [1]
	IEEE 828 14.2.2	Define release policy [1]
	IEEE 828 8.2.5.4	Establish criteria for baselines [1]
	IEEE 828 8.2.5.7	Establish change control process [1, 2]
	IEEE 828 8.2.5.9	Establish CM procedures [1]
	IEEE 828 9.2.1.1	Designate items subject to change control [1]
	IEEE 828 9.2.3	Establish change request form [1]
	IEEE 828 9.2.4.2	Control changes to baselines [1]
	IEEE 828 14.2.1	Delineate general requirements [1]
	IEEE 828 14.2.3	Define release planning [1]

A.2.3 Information item management

Name	Information item management
Purpose	The purpose of the information item management process is to develop and maintain the recorded information produced by CM processes.
Outcomes	As a result of successful implementation of this process: 1. information items are produced in accordance with defined criteria 2. information items are controlled and issued according to defined criteria 3. information items are communicated to affected parties 4. information items are maintained in accordance with planned arrangements 5. the integrity of information items is assured

Requirements traceability	IEEE 828 6.2.1.3	Document CMP [1, 2]
	IEEE 828 7.2.2.5	Update plans [3, 4]
	IEEE 828 7.2.2.1	Monitor resource usage [4]
	IEEE 828 9.2.5.3	Report disposition of change requests [3]
	IEEE 828 10.2.1	Verify CI status information needs [4]

A.3 Statement of conformity to ISO/IEC 15504-2

A.3.1 General

The process reference model (PRM) described in A.1.1 and A.1.2 is suitable for use in process assessment performed in accordance with ISO/IEC 15504-2. ISO/IEC 15504-2, Clause 6.2 places requirements on process reference models suitable for assessment against ISO/IEC 15504-2. The following subclauses quote the requirements for a PRM and describe how this standard meets these. In each of the following clauses, the text in a box quotes the requirements from the text of ISO/IEC 15504-2, and the text below each box describes the manner in which the requirements are satisfied in this standard.

A.3.2 Requirements for process reference models

ISO/IEC 15504-2:2003, Information technology—Process assessment—Performing an assessment

14.2.8.1 A Process Reference Model shall contain:

a) A declaration of the domain of the Process Reference Model.

b) A description, meeting the requirements of Clause 6.2.4 of this International Standard, of the processes within the scope of the Process Reference Model.

c) A description of the relationship between the Process Reference Model and its intended context of use.

d) A description of the relationship between the processes defined within the Process Reference Model.

— The domain of the process reference model (PRM) is Configuration Management in Systems and Software Engineering, as described in Clause 1 of this standard.

— The description of the processes is provided in A.1.1 and A.1.2 of this standard.

— This PRM is a logical representation of the elements of the processes within the domain of configuration management in systems and software engineering. Using the PRM in a practical application may require additional processes from ISO/IEC/IEEE 12207™ to suit the environment and circumstances. The PRM describes at an abstract level the processes implied by the requirements contained in Clause 6 to Clause 14. Each process of this PRM is described in terms of a purpose and outcomes. The PRM does not attempt to place the processes in any specific environment nor does it pre-determine any level of process capability required to satisfy the requirements in Clause 6 to Clause 14 of this standard.

— A description of the relationship between the processes defined within this PRM is supported by Figure 2. A.1.1.1 identifies the differences between the processes described in A.1.1, A.1.2, and those identified in Figure 2.

ISO/IEC 15504-2: 2003, Information technology—Process assessment—Performing an assessment

6.2.3.2 The Process Reference Model shall document the community of interest of the model and the actions taken to achieve consensus within that community of interest:

a) The relevant community of interest shall be characterized or specified.

b) The extent of achievement of consensus shall be documented.

c) If no actions are taken to achieve consensus a statement to this effect shall be documented.

— The relevant communities of interest and their mode of use are described in Clause 4 of this standard.

— The consensus requirements of the IEEE are satisfied by the publication of this standard.

— No actions are required because consensus has been achieved.

ISO/IEC 15504-2: 2003, Information technology—Process assessment—Performing an assessment

6.2.3.3 The processes defined within a Process Reference Model shall have unique process descriptions and identification.

— The process descriptions are unique. The identification is provided by unique names and by the identifier of each process of this standard.

A.3.3 Process descriptions

ISO/IEC 15504-2: 2003, Information technology—Process assessment—Performing an assessment

14.2.4 The fundamental elements of a Process Reference Model are the descriptions of the processes within the scope of the model. The process descriptions in the Process Reference Model incorporate a statement of the purpose of the process which describes at a high level the overall objectives of performing the process, together with the set of outcomes which demonstrate successful achievement of the process purpose. These process descriptions shall meet the following requirements:

a) a process shall be described in terms of its purpose and outcomes;

b) in any process description the set of process outcomes shall be necessary and sufficient to achieve the purpose of the process;

c) process descriptions shall be such that no aspects of the Measurement Framework as described in Clause 5 of this International Standard beyond level 1 are contained or implied.

An outcome statement describes one of the following:

— Production of an artifact;

— A significant change of state;

— Meeting of specified constraints, e.g. requirements, goals etc.

— These requirements are met by the process descriptions in A.1.1 and A.1.2 of this standard.

Annex B

(informative)

Mapping IEEE Std 828 to ISO/IEC/IEEE 12207:2008

This table identifies linkages between the subclauses of IEEE Std 828 and ISO/IEC/IEEE 12207:2008. A link should be interpreted as one or more concerns (or requirements) that are common to the identified subclauses.

IEEE Std 828		ISO/IEC/IEEE 12207:2008	
Identify management information needs	6.2.1.1	6.3.5.3.1	Configuration Management: Planning
Determine reporting needs	6.2.1.1.1		
Determine reporting frequency	6.2.1.1.2		
Identify information needed to manage CM activities	6.2.1.2	7.2.2.3.1.1	Configuration Management Process: Process implementation: Establish plan
Document CMP	6.2.1.3	7.2.2.3.1.1	Configuration Management Process: Process implementation: Establish plan
Manage implementation of CMP	7.2.1		
Monitor resource usage	7.2.2.1		
Monitor progress	7.2.2.2		
Monitor risks	7.2.2.3		
Identify variances	7.2.2.4		
Update plans	7.2.2.5		
Establish the structure and hierarchy of CIs	8.2.1	7.2.2.3.2.1	Configuration Management Process: Configuration identification
Identify CIs	8.2.2	6.1.1.3.6.3 6.3.5.3.1 6.4.03.3.1.1 7.2.2.3.2.1	Acquirer acceptance: Configuration management following acceptance Configuration management: Planning System architectural design: Establish the system architecture design Configuration Management Process: Configuration identification
Describe CIs	8.2.3	6.3.5.3.1 6.4.03.3.1.1 7.2.2.3.2.1	Configuration management: Planning System architectural design: Establish the system architecture design Configuration Management Process: Configuration identification
Establish naming convention	8.2.4.1	7.2.2.3.2.1	Configuration Management Process: Configuration identification
Establish controlled repositories	8.2.5.1		
Acquire electronic CIs	8.2.5.2		
Acquire physical CIs	8.2.5.3	6.3.5.3.1 6.4.03.3.1.1 7.2.2.3.2.1	Configuration management: Planning System architectural design: Establish the system architecture design Configuration Management Process: Configuration identification
Establish criteria for baselines	8.2.5.4		
Define how baselines are established	8.2.5.5		

IEEE Std 828		ISO/IEC/IEEE 12207:2008	
Identify baselines	8.2.5.6	6.3.5.3.1 6.4.03.3.1.1 7.2.2.3.2.1 7.2.2.3.3.1	Configuration management: Planning System architectural design: Establish the system architecture design Configuration Management Process: Configuration identification Configuration Management Process: Configuration control
Establish change control process	8.2.5.7	6.1.1.3.6.3	Acquirer acceptance: Configuration management following acceptance
Establish physical storage procedures	8.2.5.8	7.2.1.3.4.1	Documentation Process: Maintenance
Establish CM procedures	8.2.5.9	7.2.1.3.4.1	Documentation Process: Maintenance
Assure system maintenance for CM repository reliability	8.2.5.10		
Designate items subject to change control	9.2.1.1		
Establish change evaluation criteria and authorities (CCB)	9.2.2	6.1.1.3.6.3	Acquirer acceptance: Configuration management following acceptance
Establish change request form	9.2.3		
Control changes to constituent CIs	9.2.4.1	7.2.2.3.3.1	Configuration Management Process: Configuration control
Control changes to baselines	9.2.4.2		
Verify implementation of approved changes	9.2.5.1	7.2.2.3.3.1	Configuration Management Process: Configuration control
Verify non-implementation of deferred/rejected items	9.2.5.2		
Report disposition of change requests	9.2.5.3		
Verify CI status information needs	10.2.1	6.3.5.3.2	Configuration management: Execution
Verify CM mechanisms are properly set up to support information needs	10.2.2		
Account for all CIs	10.2.3	6.3.5.3.2	Configuration management: Execution
Report on status of CIs	10.2.4	7.2.2.3.4.1	Configuration Management Process: Configuration status accounting
Report discrepancies from audits	10.2.5	7.2.2.3.4.1	Configuration Management Process: Configuration status accounting
Perform functional configuration audits	11.2.1		
Inspect traceability	11.2.1.1	7.2.2.3.3.1 7.2.2.3.5.1	Configuration Management Process: Configuration control Configuration Management Process: Configuration evaluation
Perform physical configuration audits	11.2.2		
Inspect physical attributes	11.2.2.1	7.2.2.3.3.1 7.2.2.3.5.1	Configuration Management Process: Configuration control Configuration Management Process: Configuration evaluation
Perform baseline configuration audit	11.2.3		
Inspect traceability	11.2.3.1	7.2.2.3.3.1 7.2.2.3.5.1	Configuration Management Process: Configuration control Configuration Management Process: Configuration evaluation
Record nonconformities	11.2.4		
Verify discrepancy resolution	11.2.5		
Identify the product's key interfaces	12.2.1		
Control the interface specifications	12.2.2	6.1.1.3.6.3	Acquirer acceptance: Configuration management following acceptance
Activities and tasks	13.2		
Include handling of acquired items in the CMP	13.2.1	6.3.5.3.1	Configuration management: Planning

IEEE Std 828		ISO/IEC/IEEE 12207:2008	
Place acquired items under CM	13.2.2		
Delineate general requirements	14.2.1		
Define release policy	14.2.2		
Define release planning	14.2.3		
Define release contents	14.2.4	6.3.5.3.1 6.4.03.3.1.1 7.2.2.3.2.1	Configuration management: Planning System architectural design: Establish the system architecture design Configuration Management Process: Configuration identification
Define release format and distribution	14.2.5		
Define release tracking	14.2.6		
Deliver approved releases	14.2.7		
Archive	14.2.8		

Annex C

(informative)

Mapping IEEE Std 828 to ISO/IEC/IEEE 15288:2008

This table identifies linkages between the subclauses of IEEE Std 828 and ISO/IEC/IEEE 15288:2008. A link should be interpreted as one or more concerns (or requirements) that are common to the identified subclauses.

IEEE Std 828		ISO/IEC/IEEE 15288:2008	
Identify management information needs	6.2.1.1	6.3.5.1	Configuration Management: Plan configuration management
Determine reporting needs	6.2.1.1.1		
Determine reporting frequency	6.2.1.1.2		
Identify information needed to manage CM activities	6.2.1.2		
Document CMP	6.2.1.3		
Manage implementation of CMP	7.2.1		
Monitor resource usage	7.2.2.1		
Monitor progress	7.2.2.2		
Monitor risks	7.2.2.3		
Identify variances	7.2.2.4		
Update plans	7.2.2.5		
Establish the structure and hierarchy of CIs	8.2.1		
Identify CIs	8.2.2	6.3.5.1	Configuration Management: Plan configuration management
Describe CIs	8.2.3	6.3.5.1	Configuration Management: Plan configuration management
Establish naming convention	8.2.4.1		
Establish controlled repositories	8.2.5.1		
Acquire electronic CIs	8.2.5.2		
Acquire physical CIs	8.2.5.3	6.3.5.1	Configuration Management: Plan configuration management
Establish criteria for baselines	8.2.5.4		
Define how baselines are established	8.2.5.5		
Identify baselines	8.2.5.6	6.3.5.1	Configuration Management: Plan configuration management
Establish change control process	8.2.5.7		
Establish physical storage procedures	8.2.5.8		
Establish CM procedures	8.2.5.9		
Assure system maintenance for CM repository reliability	8.2.5.10		
Designate items subject to change control	9.2.1.1		
Establish change evaluation criteria and authorities (CCB)	9.2.2		
Establish change request form	9.2.3		
Control changes to constituent CIs	9.2.4.1	6.3.5.2	Configuration Management: Perform configuration management
Control changes to baselines	9.2.4.2		
Verify implementation of approved changes	9.2.5.1	6.3.5.2	Configuration Management: Perform configuration management
Verify non-implementation of deferred/rejected items	9.2.5.2		

IEEE Std 828		ISO/IEC/IEEE 15288:2008	
Report disposition of change requests	9.2.5.3		
Verify CI status information needs	10.2.1	6.3.5.2	Configuration Management: Perform configuration management
Verify CM mechanisms are properly set up to support information needs	10.2.2		
Account for all CIs	10.2.3	6.3.5.2	Configuration Management: Perform configuration management
Report on status of CIs	10.2.4		
Report discrepancies from audits	10.2.5		
Perform functional configuration audits	11.2.1		
Inspect traceability	11.2.1.1		
Perform physical configuration audits	11.2.2		
Inspect physical attributes	11.2.2.1		
Perform baseline configuration audit	11.2.3		
Inspect traceability	11.2.3.1		
Record nonconformities	11.2.4		
Verify discrepancy resolution	11.2.5		
Identify the product's key interfaces	12.2.1		
Control the interface specifications	12.2.2		
Activities and tasks	13.2		
Include handling of acquired items in the CMP	13.2.1	6.3.5.1	Configuration Management: Plan configuration management
Place acquired items under CM	13.2.2		
Delineate general requirements	14.2.1		
Define release policy	14.2.2		
Define release planning	14.2.3		
Define release contents	14.2.4	6.3.5.1	Configuration Management: Plan configuration management
Define release format and distribution	14.2.5		
Define release tracking	14.2.6		
Deliver approved releases	14.2.7		
Archive	14.2.8		

Annex D

(normative)

The configuration management plan (CMP)

The CMP shall include the following either by reference to another document that is a CI or within itself.

D.1 Introduction to the plan

Introduction information provides a simplified overview of the CM activities so that those approving, those performing, and those interacting with CM can obtain a clear understanding of the agreed CM activities. The introduction shall include these topics: the purpose of the plan, the scope of the CM effort and how it fits into the larger organization, key terms and their definitions, and references.

D.1.1 Purpose of the plan

The purpose shall briefly address why the plan exists and describe the intended audience.

D.1.2 Scope of the plan

The scope shall address CM applicability, limitations, and assumptions on which the plan is based.

D.1.3 Relationship to the organization and other projects

The overview provides a description of the product and portion of the product life cycle served by this CM effort, degree of formality of engineering processes, depth of CM control, and deliverables to be released at project close.

D.1.4 Key terms

Key terms shall be defined as they apply to the plan in order to establish a common terminology among all users of the plan. The CMP shall include a glossary or provide a reference to a project glossary.

D.1.5 References

All references in the plan to policies, directives, procedures, standards, terminology, and related documents shall be uniquely identified to aid retrieval by users of the plan.

D.2 Criteria for identification of the configuration items (CIs) to which CM will be applied

What types of items will be subject to CM shall be defined, along with criteria for entry into CM repositories.

D.3 Limitations and assumptions affecting the plan

The CMP should describe limitations, such as time constraints, that apply to the plan and assumptions that might have an impact on the cost, schedule, or ability to perform defined CM activities (e.g., assumptions of the degree of customer participation in CM activities or the availability of automated aids).

D.4 CM responsibilities and authorities

CM management information describes the allocation of responsibilities and authorities for CM activities and their management, as described in Clause 7 of this standard.

CM management information shall include four topics: the project organization(s) within which CM is to apply, the CM responsibilities of these organizations, references to the CM policies and directives that apply to this project, and the management of the CM process.

D.5 Project organization

The organizational context, both technical and managerial, within which the planned CM activities are to be implemented, shall be described. The plan shall identify the following:

a) All organizational units that participate in or are responsible for any CM activity on the project

b) All organizational units that participate in or are responsible for the problem resolution

c) The functional roles of these organizational units within the project structure

d) Relationships between organizational units and the interfaces implementing the relationships

e) Organizational units may consist of a vendor and customer, a prime contractor and subcontractors, or different groups within one organization. Organization charts, supplemented by statements of function, roles, and relationships, can be an effective way of presenting this information.

D.6 CM responsibilities

The allocation of CM activities to organizational units shall be specified. For each activity listed within CM activities, the name of the organizational unit or job title to perform this activity shall be provided. A matrix that relates the organizations defined above to the CM functions, activities, and tasks can be useful for documenting the CM responsibilities.

For any review board or special organization established for performing CM activities on this project, the plan shall describe its:

a) purpose and objectives

b) membership and affiliations

c) period of effectivity

d) scope of authority

e) operational procedures

D.7 Applicable policies, directives, and procedures

Any organizational policies, management guidance and directives, and procedures that apply to the planned effort shall be identified. For each, its impact and effect on the plan shall be stated.

D.8 Planned activities, schedule and resources

The organizational unit responsible for the overall CM process shall specify CM activities for the project in the SCMP. The plan shall explain how the organization will perform the required CI, change control, status accounting, and auditing activities in Clause 8 through Clause 12 of this standard, and may also cover the interface, supplier, and release management activities of Clause 12, Clause 13, and Clause 14.

D.8.1 CM schedules

CM schedule information establishes the sequence and coordination for the identified CM activities and for all events affecting the plan's implementation.

The plan shall state the sequence and dependencies among all CM activities and the relationship of key CM activities to project milestones or events. The schedule shall cover the duration of the plan and contain all major milestones of the project related to CM activities. Milestones shall include establishment of a configuration baseline, implementation of change control procedures, and the start and completion dates for a configuration audit.

Schedule information shall be expressed as absolute dates, as dates relative to either CM or project milestones, or as a simple sequence of events.

D.8.2 CM resources

CM resource information identifies the personnel, environment, infrastructure, tools, techniques, equipment, and training necessary for the implementation of the specified CM activities.

For each type of CM activity identified, the plan shall specify what personnel, tools, techniques, equipment, and training are required and how each resource will be provided or obtained.

D.9 CMP maintenance

CMP maintenance information identifies the activities and responsibilities necessary to ensure continued CM planning during the life cycle of the project or service. The plan should be reviewed at the start of each project phase (or appropriate CM services interval), changed accordingly, and approved and distributed to the project team. The plan shall include a history of changes made to the plan and state the following:

a) Who is responsible for monitoring the plan

b) How frequently updates are to be performed

c) How changes to the plan are to be evaluated and approved

d) How changes to the plan are to be made and communicated

Annex E

(informative)

Examples of how CM planning and management are applied

E.1 Requirements

As soon as the project is launched, the CM authority assures that the CMP is sufficiently detailed to support establishment of CM processes, policies, and procedures, recognizing that additional details will be supplied before they are needed as the project evolves. The CM manager works with the project manager to identify points in the project schedule at which additional detail will be required. The CM authority works with the project manager to assure that the CM environment and tools are properly installed and configured. The CM authority meets with the identified resources within the groups (e.g., systems engineering, software development, system testing) who have been assigned CM tasks or responsibilities or with whom a CM group interfaces to assure that roles and responsibilities are well understood. If training is required for these resources, the CM authority will work with the project manager to have it provided. CM processes should be audited to determine if requirements configuration items are being correctly handled and deficiencies recorded and addressed.

E.2 Design

The CM authority should ensure that software developers on the project understand the selected branching model and how to use the toolset to implement it. CM processes should be audited to determine if design configuration items are being correctly handled and deficiencies recorded and addressed.

E.3 Construction and integration

The CM authority should verify that software developers on the project are using the CM toolset appropriately and using branches appropriately. CM processes should be audited to determine if software configuration items are being correctly handled and deficiencies recorded and addressed.

E.4 Qualification testing

The CM authority should verify that change requests (CRs) arising from testing efforts are being properly reported and tracked to disposition, so that any changes in the product (code, documentation, etc.) can be traced back to the initiating CR. The CM authority should verify that releases to any external testing function are produced as planned. Any deficiencies should be recorded and addressed.

E.5 Installation and acceptance

The CM authority should verify that products released for installation and acceptance are produced using the appropriate CM processes and that the formal software portions of the release are held in a Definitive Media Library and is the only one that is installed for acceptance testing and deployment. Any deficiencies should be recorded and addressed.

E.6 Operation

For software components, the CM authority should verify that all code in production exactly matches the code in the Definitive Media Library. Any deficiencies should be recorded and addressed.

E.7 Maintenance

The CM authority should verify that any change made to production code has been approved by the configuration control board. Any deficiencies should be recorded and addressed.

E.8 Disposal

The CM authority should verify that the system designated for retirement has been completely removed from the operating (production) environment and archived. Any deficiencies should be recorded and addressed.

Annex F

(informative)

Examples of how configuration identification (CI) is applied

F.1 Requirements

During requirements analysis, the technical requirements are derived from the established business needs and documented typically in an iterative fashion.

Individual requirements, once baselined, become configuration items. The sets of requirements baselined at the close of iterations also become configuration items to be managed.

Requirements may be captured in the form of a specification, feature specification, user stories, use cases, prototype, user interface mock ups, etc.).

Once requirements are captured, they should be uniquely identified to enable traceability to system architecture or components of the overall system. Following incorporation of internal review comments and approval, the requirements should be baselined.

During each iteration of requirements analysis, a new requirements baseline is established. Work in each iteration may later dictate changes to the requirements.

Such instances should evoke the change control process described in Annex H.

F.2 Design

During initial design and each subsequent iteration, as the overall structure of the product emerges, CM should work with the developers during the first iteration/phase to:

a) Inform developers of any naming constraints imposed by the CM tool(s)

b) Establish an appropriate branching and merging model that assures that changes are propagated wherever required and that they are not “lost”

c) Assure that developers understand how to properly use any CM tools being used

d) Assure that developers understand tool use and CM policies used to assure the integrity of the CM repository (database)

With this information, CM should set up the CM tools with the needed policies and establish the basic branching model.

Design can be separated into architectural design (see [B12]) and component design.

This distinction is important for CM, especially in iterative and incremental development in which the high-level architectural design should be specified and baselined even before the first implementation iteration, although a component’s design occurs in the iteration in which that particular feature is implemented.

This allows all team members to have access to the same design, facilitating distributed and parallel development efforts.

Work in subsequent iterations may dictate changes to the original definition of the architecture. Such instances should invoke the change control process described below.

F.3 Construction and integration

Identification of configuration items continues throughout construction of the product as new items are developed to satisfy the product's requirements and design.

CM plays an essential role. It provides a common repository for identified items and a common process for delivery of items to the repository. Both of these are essential to effective and efficient development activities.

Effective construction relies heavily on successful coordination of cross-team activities, especially in distributed development teams, parallel development efforts, and in teams under pressure to produce working code quickly.

The most effective CM processes enable developers to

a) work in their own workspaces (all of which feature the same tools and environment) with access to the then-current product requirements, designs, and code

b) pull down the latest "approved" code into their workspaces

c) name, write, build, and test their new code in their own workspaces. Once they are confident of the new code, they then submit the changes to the CM repository (common artifact base).

Incorporation of the code (or other artifact) into the repository identifies the new CI and makes it available for the next build of the evolving product.

Once construction is complete for a delivery to an independent test organization, CM assures that the correct CIs are labeled accordingly in the repository, assures that a correct build manifest exists as a CI, and authorizes a build for external release.

F.3.1 Builds

During construction, developers/producers should perform frequent integration of changes to the evolving product.

For example, for code, builds are done nightly or even continuously.

Each build should have a unique identifier. Builds that are identified as new baselines are established as CIs.

All CIs should have an embedded, unique, and immutable version ID.

The build process should automate and verify the creation of version IDs in CIs.

F.3.2 Naming schemes

Typically, when products are released, sequence-based identifiers are used to convey the significance of changes between releases. In the following scheme, the position of the numbers in the sequence indicates the significance of the changes in the release. For example, the following sequence is commonly used: <major#>.<minor#>.<revision#>.<build#>. See Annex L for further discussion.

F.3.3 Build manifests and release notes

Each build should be accompanied by a build manifest that identifies the contents of the build. When a build is approved for baselining and release to an external organization (e.g., QA or production), the release should also be accompanied by release notes describing requirements for the installation environment, installation instructions, and a summary of what, if anything, has changed since the previous release.

F.4 Qualification testing

CM should identify the configuration items to be used in qualification testing:

a) as input for processing

b) as supporting information or tools for processing

c) as the output of processing

The items needed by the testing function may come from other engineering processes, but items to be tested should originate from the CI repository.

NOTE—Configuration items available to test managers include, but are not limited to, plans, requirements, user documentation, user method of operation, design documentation, code (object and executables), unit testware, coverage and results, runtime environments, hardware, test designs, test procedures, test scripts, test results, runtime analysis, system logs, fault reports, acceptance criteria, etc.

The test manager is responsible for analyzing the identified input configuration items and for assuring that the identified input items are received and that the identified output items are produced and delivered.

The delivered CIs include defect reports associated with CIs as well as periodic status reports describing the state of the product as tested. At a minimum, these periodic status reports should be produced when the product is eligible for promotion (e.g., from alpha to beta status, from beta to release candidate status, and from release candidate status to production status).

These status reports should become CIs.

The CM authority should carry out a sample audit of the delivered configuration items.

F.5 Installation and acceptance

CM should identify the production status CIs in the repository as the production version.

The release engineering function should provide the approved production build of the product from the repository, verify that it is properly identified as such, and deliver it in the appropriate media for installation, installation testing, and acceptance.

In software product companies, acceptance takes the form of authorization to deliver the production status "Gold Masters" to manufacturing [this process is called Release to Manufacturing (RTM)] where the product components (e.g., bill of materials, CDs, booklets, etc.) are assembled for delivery (e.g., boxed, shrink-wrapped, and delivered to Order Fulfillment).

In IT organizations, software installation involves the deployment of a specific baseline release that has been approved for promotion (i.e., accepted) by the Change Control Board. CM should confirm that the intended CIs have been installed, including modifications to any interface configuration dependencies.

Product installation should be controlled, traceable, and verifiable in terms of the change specified in the approved Request for Change (RFC).

F.6 Operation

No new CIs should be introduced during normal operation. (New CIs may be introduced during maintenance updates, however.)

F.7 Maintenance

Maintenance updates to production-level products are expected. Such updates in software may take the form of updates to tables (e.g., updates to payroll systems for changes to tax laws), updates to coordinate with Operating System upgrades, or updates to correct defects.

All such updates include the addition of new CIs to the software product.

All such updates are subject to the change control process described in Annex H.

F.8 Disposal

When the product is removed for disposal (for example, to be replaced by a newer package), it should be identified as obsolete and archived.

Annex G

(informative)

Examples of implementing change control in a software development environment

The following examples show how a configuration change control system could be implemented in a typical software development environment.

G.1 Item-level change control

When using a software configuration management tool, changes to individual files are recorded when the person making the change "commits" the changed file. This is the same type of process that happens when committing a change in any database management system. With most commit change control mechanisms, change trails are almost transparent. Attribution is obtained with the use of versioning system authentication mechanisms. The rationale for the change is supplied in the comment about the commit by the person committing the change. Notification can then be pulled from the versioning system for status accounting reports and/or triggers can be established to push the information to the relevant stakeholders (e.g., the respective technical lead or other team members). Thus, the commit record is, in fact, an audit trail. The "revert" option in the versioning system enables the project to roll back specific changes. The "diff" of the two versions is the set of changes applied.

G.2 Product-level change control and baselines

Throughout the development of a software system, a series of successive baselines enables the project team to use the same approved version. It is a software life cycle event marking the transition of the software product from one state of maturity to another. Baseline change control requires the approval of the appropriate change control board(s). The baseline change control process is a closed-loop process that assures control of the change from the change request that identifies the need for implementing the new baseline, through its approval, until the new baseline is implemented. The tracking record contains information about the set of changes required, the impact analysis, the rationale, and approval. This serves as an audit trail.

Annex H

(informative)

Examples of how configuration control is applied

H.1 Requirements

Throughout the life cycle, requirements should be managed with other CIs. Changes requested during the life cycle that affect requirements should be evaluated against cost, schedule, and impact. When a change is needed that affects requirements, the requirements documentation should be updated, and traceability for the impact should be clearly documented. Requirements changes typically require a more formal approval such as a configuration control board (CCB) to ensure that the impact is well understood in advance of any implementation of the requested change. Requirements changes often affect design decisions and code that is already in place. Requirements traceability should be maintained throughout the life cycle of the product. This can be done through the use of requirements management tools. It is vital that changes in requirements be clearly communicated to the entire project team. This is typically done through the use of CM tools to ensure that the team understands changes that are approved and the impact those changes have on the current scope of work.

H.2 Design

Like requirements CIs, current information about design CIs is critical to the entire project team because depending on an outdated design CI to write interfaces, dependent code, or tests may cause costly and time-consuming re-work for other developers, as well as for team members writing product documentation, data sheets, etc. For this reason, it is essential that changes to design CIs be governed by a responsive change management process that enables all potentially affected parties in an integrated project team to be alerted to the requested change and to have an opportunity to weigh in on the request.

H.3 Construction and integration

During initial development of a software test, stub, or feature (which may comprise more than one CI), the code remains in the developer's workspace until it has passed the development organization's internal standards for quality, which typically include passing the developer's own tests (and, in some organizations, code reviews). Only after that is it committed to the code branch for which it is intended, where it becomes a CI. Subsequent changes during construction—up to its release to an organization external to development (like QA)—do not typically go through more rigorous configuration control because the development organization's internal standards for quality should suffice.

Continuous integration embodies the low-level change control that occurs during source code development. It means that changes to the software code are checked in (submitted, committed) to the CM repository when they have passed the development organization's internal quality requirements.

Modern software development approaches adapt to changes quickly, delivering new releases via very frequent small-increment product development iterations that are potentially to be delivered outside the development organization. The software configuration change control mechanism should at a minimum safeguard the scope (requirements) of each such iteration.

H.4 Testing

Once the product has evolved sufficiently to justify release for validation and verification, the first level of formal change control is triggered. Once created, configuration items are kept under strict change control. At this point, problems that are detected are reported. (They may be called by any number of terms, like "defects," "bugs," or "incidents.") This report initiates the change control cycle for the given item. Most organizations have a tool that enables them to initiate, communicate, and resolve bug reports. (Others simply keep lists of bugs and their disposition.)

Configuration control should assure that configuration items delivered for validation and verification are the current and correct items and are properly labeled and installed in the correct environment. This change control process should be a closed-loop process; that is, it will provide mechanisms to identify a defect report or change request and trace it through until its final disposition (acceptance and implementation, revision, rejection, or deferral).

Similarly, the configuration control process should assure that any changes to configuration items used to test the product are also governed by the closed-loop process.

Once a particular release has been approved for release for production use, each element of that release should be marked and versioned to reflect its production status.

H.5 Acceptance

Once the product is approved as a candidate for production status, no further change requests will be processed against that baseline unless problems arise during acceptance testing that cause the product to be rejected. The configuration control process should assure that the proper set of CIs is delivered, including the approved release notes and installation and operation instructions.

The configuration control process should assure that any problems detected by the operator during acceptance testing are captured and reported to the producing organization. In addition, the configuration control process should assure that any rejected releases are appropriately marked as such and archived.

Upon acceptance, all configuration items should be retained as a Quality Record for the appropriate time required by organizational policy. For software, both source and object code should be archived by the producer. The accepting organization should place the original form of the software that it has accepted into a designated media library.

Annex I

(informative)

Examples of how configuration status accounting is applied

Status accounting reports should enable analysis of project status and product stability by tracking baselines established, total changes requested and implemented, rate of change per day/week/etc., change volatility per CI, etc.

I.1 Requirements

Throughout the life cycle, requirements CIs will change. Each change should be traceable back to a requirement, to a defect record, or to a request for change. Metrics that track and measure requirements may also help the team identify project performance trends and improvement opportunities. Some of these metrics could be:

— distribution of requirements over releases

— number of requirements changes requested/accepted/deferred in any given release

I.2 Design

Typically, design CIs are associated with features, and one way that progress is tracked is by tracking the progress of each feature. That is, as feature design CIs are recorded, the team can get an idea of how much of the entire product's design has been accomplished. Further, CM status accounting can report on the volatility of specific design CIs. Especially volatile design CIs may, for example, reflect an underlying problem with requirements or with the architectural design.

I.3 Construction and testing

During construction and testing, CI status accounting keeps track of each of the current CIs, the current configuration of the product as a whole, past versions, and the status of changes to those items, from change request or defect report through change disposition, whether implemented, rejected, deferred, or pending. Status accounting should result in regular (preferably frequent) status reports to the project team.

The reports should include information such as

a) Total number of changes requested

b) Total number of changes implemented

c) Rate of change per day/week/etc.

d) Change volatility per CI, etc.

e) Total number of defects reported, by type (such as defect severity or defect source component)

f) Total number of defects repaired, deferred, pending, etc., by type

Such information provides objective insights into how the project is doing in terms of creating stable artifacts, what types of problems are occurring to slow the development and delivery process, and which components are potentially problematic (error-prone).

Status accounting and reported information should also be available to describe and track the build, release, and delivery information that is necessary for the formal control of the release and delivery activity. Status accounting for builds should include the status of each build, whether successful or not, as well as its life cycle status (development build, QA build, release build). Status accounting and reporting can also include information about build issues caused by CIs and issues arising from faulty configuration hand-offs to supporting organizations, like QA.

I.4 Acceptance

Acceptance should rely on release status reporting to determine if the release criteria are met in the product put forward for acceptance (e.g., for release into a production environment or release to a customer). Release criteria may include such measures as:

a) All high-severity defects have been corrected

b) Previously identified error-prone modules have become stable

Thus, at the beginning of the project, CM should verify that the data elements required by the release criteria can be captured and reported on. Acceptance managers should ensure that there is a robust process in place defining how this release status reporting is to be carried out and by whom.

Where possible, some form of automation in this area is desirable to facilitate frequent and accurate status reporting throughout the project, and well before the acceptance gate, so that acceptance goes smoothly.

I.5 Maintenance

Maintenance involves the management of changes to CIs operating in a production (operational) environment. Changes may be required to repair defects and to modify functional characteristics, such as compatibility with a new operating system or to become current with new external constraints (for example, payroll systems must be updated annually to reflect changes in tax laws). Thus, a disciplined and orderly approach in which requested changes are approved by a CCB, and then planned and released within the application life cycle, should be defined and exercised. Changes should be deployed in a way that allows for them to be traceable and retractable (e.g., backed out) if necessary. Maintenance changes should be traceable back to the initiating Request for Change (RFC), defect report, or documented requirement.

Release status reporting during maintenance events enables assurance that the production system is under control and that no unexpected maintenance changes have a chance of interrupting operations. Thus, status reporting should be frequent and should be sufficient to:

a) Assure that no unapproved changes have been introduced into the operating environment

b) Assure that approved changes have been implemented as intended

c) Provide visibility into the status of any requested and pending changes

d) Provide insight into the probable impact of pending changes

I.6 Operations

Operations involve the management of CIs in the runtime environment. Operations staff should regularly verify that the correct versions of all CIs are in place and immediately identify any and all unauthorized changes, if unauthorized changes occur.

Regular operations status reports should address whether:

a) Any unapproved changes were introduced into the operating environment during the reporting period

b) Approved changes are operating as expected

c) Known pending changes are expected to adversely affect operations

Annex J

(informative)

Examples of how configuration auditing is applied

J.1 Requirements

Audits are performed throughout the whole life cycle, after the first baseline has been established. Audits are performed at the end of iterations or phases, depending on the software and systems development approach. Each configuration audit also audits the CIs of all previous iterations or phases.

J.1.1 Physical configuration audit

Physical configuration audits (PCAs) of the outputs of the requirements process in the life cycle verify the following:

a) Requirements assets have been placed under configuration control

b) Requirements assets have been properly labeled in accordance with the CMP

c) An inventory of requirements assets exists and correctly reflects the attributes of each CI

d) There is evidence of the use of change control procedures for each of the changes made to previous baselines, if any (for example, in a previous iteration)

J.1.2 Functional configuration audit

Functional configuration audits (FCAs) focus on ensuring the integrity between the different models of the product throughout the life cycle. In waterfall development environments, there may be no need for an FCA at the close of the requirements phase, as this is the first model developed. In iterative development approaches, each iteration may produce new requirements or refinements to existing requirements, so FCAs of successive models are required to verify integrity.

J.2 Design

J.2.1 Physical configuration audit

PCA of the outputs of the design process verifies the following:

a) Design assets have been placed under configuration control

b) Design assets have been properly labeled in accordance with the CMP

c) An inventory of design assets exists and correctly reflects the attributes of each CI

d) There is evidence of the use of change control procedures for each of the changes made to previous baselines, if any (for example, in a previous iteration or a previous phase)

J.2.2 Functional configuration audit

FCA of the outputs of the design process verifies the following:

a) Traceability between the design items and their sources (requirements)

b) Every requirement is linked to at least one design element

c) Every design element is linked to at least one requirement that justifies it

J.3 Construction and integration

J.3.1 Physical configuration audit

PCA of the outputs of the construction and integration verifies the following:

a) Source code and related coding assets have been placed under configuration control

b) An inventory of source code and related assets correctly reflects the attributes of each CI

c) Code branches and merges follow the CMP

d) Labeling of the source code and related assets follow the CMP

e) There is evidence of the use of change control procedures for each of the changes made to previous baselines (for example, in a previous iteration or a previous phase)

f) Builds have been performed in accordance with the CMP in the following aspects:

   1. Build sequence and procedure
   2. Bill of materials
   3. Build environment
   4. Build reproducibility
   5. Build identification (both input and output) and versioning

J.3.2 Functional configuration audit

FCA of the outputs of the construction and integration verifies traceability between the source code and related items and their source (design and, indirectly, requirements assets):

a) Every design element has been transformed into source code or a related coding asset

b) Each source code file and related coding asset has at least one design element that justifies it

J.4 Qualification testing

J.4.1 Physical configuration audit

PCA of the outputs of the qualification testing process integration verifies the following:

a) Test cases and related qualification testing assets have been placed under configuration control

b) An inventory of test cases and related assets correctly reflects the attributes of each CI

c) Test cases and related assets have been labeled in accordance with the CMP

d) There is evidence of the use of change control procedures for each of the changes made to previous baselines (for example, in a previous iteration or a previous phase)

J.4.2 Functional configuration audit

FCA of the outputs of the qualification testing verifies:

a) Traceability between the test cases and related items and their source (requirements)
    1. Every requirement (and optionally design element) is mapped to at least one test case
    2. Each test case has at least one requirement (and optionally design element) that justifies it
b) Traceability between the test executions and the test cases
    1. Every test case defined was executed at least once to verify that the requirement was implemented
    2. Every test failed has been analyzed and re-run if a repair has been implemented, or the defect is reported as an accepted known issue

J.5 Installation and acceptance

J.5.1 Physical configuration audit

PCA of outputs of the installation and acceptance verifies that:

a) Installation and acceptance assets have been placed under configuration control
b) An inventory of installation and acceptance assets correctly reflects the attributes of each CI
c) Installation and acceptance assets have been labeled in accordance with the CMP
d) There is evidence of the use of change control procedures for each of the changes made to previous baselines (for example, in a previous iteration or a previous phase)
e) Release notes are correct
    1. Identification of the product version
    2. List of implemented features
    3. List of known defects
    4. Installation procedures/guidance

J.5.2 Functional configuration audit

FCA of the results of the installation and acceptance is usually a corroboration of the whole functional configuration of the previous phases since this phase is not intended to transform the product in any way.

J.6 Maintenance

Maintenance implies, from the configuration management point of view, a complete cycle of development, which makes both PCA/FCA after each maintenance cycle an aggregated version of the PCAs/FCAs of pre-release phases, with special emphasis on the correctness of the change control process of the maintenance request performed.

Annex K

(informative)

Software build naming schemes

Sequence-based identifiers are used to name each software build. The choice of sequence numbers conveys the significance of changes between releases: <major#>.<minor#>.<revision#>.<build>.

The first set in the sequence is changed only when the release contains major functional changes to the product. Changes to sets in the sequence after the first represent changes of decreasing significance. In other words, the major number is increased when there are significant changes in functionality, the minor number is incremented for smaller changes or important bug fixes, and the revision number is incremented for minor bugs fixes.

It is common to use a series like the following:

- For pre-release of a new software product: 0.9, with revisions labeled 0.9.1, 0.9.2, 0.9.3, etc.
- For the first major release: 1.0, followed by, for example, 1.0.1, 1.0.2, 1.1, 1.1.1, etc.
- For the second major release: 2.0, 2.0.1, 2.0.2, 2.1, 2.1.1, 2.1.2, 2.2, etc.

Each build during internal product development shall follow a pre-defined naming scheme.

In some organizations, the build number alone is used. In others, alphabetic discriminators are added. For example:

- Development builds may carry the discriminator “d”
- Builds delivered for testing purposes may be labeled “a” (for alpha releases) or “b” (for beta releases)
- Builds delivered as release candidates carry the discriminator “rc”
- An example series for the second major release of a product evolving through the development and delivery cycle would be: 2.0.0.xdx (development builds), 2.0.0.xax, 2.0.0.xbx, 2.0.0.xrcx

NOTE—The integrity that this naming discipline assures assumes a prohibition against making any changes at all when going from the last “d” build to the first “a” build, and so on, from the last beta to the first release candidate, and from the last release candidate to production.

Annex L

(informative)

Mapping IEEE Std 828 to ISO 10007:2003

This table identifies linkages between the sub-clauses of IEEE Std 828 and ISO 10007. A link should be interpreted as one or more concerns (or requirements) that are common to the identified subclauses.

IEEE Std 828		ISO 10007:2003	
Identify management information needs	6.2.1.1	5.5.1	General
Determine reporting needs	6.2.1.1.1		
Determine reporting frequency	6.2.1.1.2		
Identify information needed to manage CM activities	6.2.1.2		
Document CMP	6.2.1.3		
Manage implementation of CMP	7.2.1		
Monitor resource usage	7.2.2.1		
Monitor progress	7.2.2.2		
Monitor risks	7.2.2.3		
Identify variances	7.2.2.4		
Update plans	7.2.2.5		
Establish the structure and hierarchy of CIs	8.2.1	5.3.2	Product configuration information
Identify CIs	8.2.2	5.3.1 5.3.2	Product structure and selection of configuration items Product configuration information
Describe CIs	8.2.3		
Establish naming convention	8.2.4.1	5.3.2	Product configuration information
Establish controlled repositories	8.2.5.1	5.5.2.3	Product configuration environment
Acquire electronic CIs	8.2.5.2		
Acquire physical CIs	8.2.5.3		
Establish criteria for baselines	8.2.5.4		
Define how baselines are established	8.2.5.5		
Identify baselines	8.2.5.6	5.3.3	Configuration baselines
Establish change control process	8.2.5.7	5.4.1 5.4.4	General Disposition of change
Establish physical storage procedures	8.2.5.8		
Establish CM procedures	8.2.5.9		
Assure system maintenance for CM repository reliability	8.2.5.10		
Designate items subject to change control	9.2.1.1	5.4.1 5.4.2	General Initiation, identification and documentation of the need for change
Establish change evaluation criteria and authorities (CCB)	9.2.2	5.4.1 5.4.2	General Initiation, identification and documentation of the need for change
Establish change request form	9.2.3	5.4.2	Initiation, identification and documentation of the need for change
Control changes to constituent CIs	9.2.4.1	5.4.1 5.4.2 5.4.4	General Initiation, identification and documentation of the need for change Disposition of change

IEEE Std 828		ISO 10007:2003	
Control changes to baselines	9.2.4.2		
Verify implementation of approved changes	9.2.5.1	5.4.5	Implementation and verification of change
Verify non-implementation of deferred/rejected items	9.2.5.2		
Report disposition of change requests	9.2.5.3		
Verify CI status information needs	10.2.1	5.5.2.1 5.5.2.2 5.5.3	Record content Record tracebility information Reports
Verify CM mechanisms are properly set up to support information needs	10.2.2		
Account for all CIs	10.2.3	5.5.2.1 5.5.2.2 5.5.3	Record content Record traceability information Reports
Report on status of CIs	10.2.4		
Report discrepancies from audits	10.2.5		
Perform FCAs	11.2.1	5.6	Configuration audit
Inspect traceability	11.2.1.1	5.6	Configuration audit
Perform PCAs	11.2.2	5.6	Configuration audit
Inspect physical attributes	11.2.2.1	5.6	Configuration audit
Perform baseline configuration audit	11.2.3	5.6	Configuration audit
Inspect traceability	11.2.3.1	5.6	Configuration audit
Record nonconformities	11.2.4		
Verify discrepancy resolution	11.2.5		
Identify the product's key interfaces	12.2.1		
Control the interface specifications	12.2.2		
Activities and tasks	13.2		
Include handling of acquired items in the CMP	13.2.1	5.5.1	General
Place acquired items under CM	13.2.2		
Delineate general requirements	14.2.1		
Define release policy	14.2.2		
Define release planning	14.2.3		
Define release contents	14.2.4		
Define release format and distribution	14.2.5		
Define release tracking	14.2.6		
Deliver approved releases	14.2.7		
Archive	14.2.8		

Annex M

(informative)

Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

[B1] *The IEEE Standards Dictionary: Glossary of Terms & Definitions*, Standards Information Network, IEEE Press, 2008.

[B2] ISO 9660:1988, Information Processing–Volume and Structure of CD-ROM for Information Interchange.

[B3] ISO/IEC/IEEE 12207™:2008, Software and Systems Engineering–Software Life-Cycle Processes.

[B4] ISO/IEC/IEEE Standard 15288:2008, IEEE Standard for Systems and Software Engineering–System Life-Cycle Processes.

[B5] ISO/IEC/IEEE 15289:2011, Systems and Software Engineering–Content of Life-Cycle Information Products.

[B6] ISO/IEC 15939:2007, Software and System Engineering–Measurement Process.

[B7] ISO/IEC 16085:2006, Information Technology–Systems & Software–Life-Cycle Processes–Risk Management.

[B8] ISO/IEC 16326: 2009, Systems and Software Engineering–Life-Cycle Processes–Project Management.

[B9] ISO/IEC 19770-1:2006, Information Technology–Software Asset Management–Part 1: Processes.

[B10] ISO/IEC 20000-1:2011, Information Technology–Service Management–Part 1: Service Management System Requirements.

[B11] ISO/IEC/IEEE 24765™:2009, Systems and Software Engineering–Vocabulary.

[B12] ISO/IEC 42010:2007, Systems and Software Engineering–Recommended Practice for Architectural Description of Software-Intensive Systems.

[B13] ISO/IEC TR 24774:2010, Systems and Software Engineering–Life-Cycle Management–Guidelines for Process Description.

[B14] ISO/IEC TR 90005:2008, Systems Engineering–Guidelines for the Application of ISO 9001 to System Life-Cycle Processes.