working simultaneously on multiple version updates. Errors or features of the software may only be present in certain versions, so it is important to be able to retrieve and run different versions of the software to determine in which version(s) the problems occur.

Software Developers could retain multiple copies of the different versions of a program, and label them appropriately, but this simple approach is risky as many near-identical copies of the program have to be maintained and it often leads to mistakes. Granting read-write-execute permission to a group of Developers is also necessary; this adds the pressure of someone managing permissions so that the code base is not compromised. Consequently, systems to automate some or all of the revision control process are available to ensure that the majority of effort in managing version control is efficient and hidden.

6.3.3 Software Configuration Identification

Configuration Identification consists of identifying the software products to be placed under configuration control. These software products are the development products identified in the SDP, the hardware and software in the Software Development Environment, plus other documents and data as required by the contract.

Program Unique Identifiers. The output of Configuration Identification is a configuration controlled list of configuration items. Each software requirement, or product, must be *uniquely identified* with a *Program Unique Identifier* (PUI) numbering system. Massive confusion will engulf a large software-intensive program if it lacks a carefully controlled identification system. This is not a trivial issue.

The Chief Software Engineer, or designee, should be responsible for ensuring that a common, and unique, software PUI scheme is used across the entire program or project. The identification scheme must be at the level at which the software entities will be controlled, for example, computer files, electronic media, documents, SIs, SUs and hardware elements.

Computer Assisted Software Engineering (CASE) Tools. CASE tools, used to support Configuration Identification, should be identified as well as how their features (e.g., versioning, branching, and labeling) are used to track and control promotion and delivery of deliverable items. CASE tools are discussed in Chapter 9.

6.3.4 Software Configuration Control

Software configuration control is the systematic control of modifications to baselined products throughout the product's life cycle. The SDP or the SCMP (see below) must describe the SCM process for controlling baselined products and establishing common SCM change procedures. Subsystem SCM procedures may be provided in the Subsystem SDP Annexes. The policies and process for approving and implementing changes to baselined software products must be defined in the SDP or SCMP. If needed, more detailed *operational procedures* can augment the direction provided in the SCMP.

Software Configuration Management Plan. The SCMP documents the policies and procedures for conducting required Software Configuration Management for all SIs. The SCMP establishes the plan for creating and maintaining a uniform system of configuration identification, control, status accounting, and audit for the software work products throughout the software development process. The SCMP can be organized into five sections:

- *Section 1 Introduction:* Presents and defines the scope and purpose of the SCMP.
- *Section 2 Applicable Documents:* Lists the compliance document(s) and other documents that are referenced, or related to, the SCMP.
- *Section 3 Organization and Resources:* Describes the overall structure of the Software CM Organization, personnel and resources to be employed.
- *Section 4 Software Configuration Management Activities:* Covers details of the major Software CM functions and activities (a major portion of the SCMP).
- *Section 5 Glossary:* Lists the abbreviations and acronyms.

Objectives of the SCMP. The objective of the SCMP is to define the process to be used by SCM personnel in managing the configuration control of the software work products. Specifically, the SCMP should provide the following guidelines, direction and/or procedures in order to:

- Identify the software development items to be baselined.
- Provide change control and visibility of the changes made to software work products through the configuration control procedures.
- Control incorporation of all approved software changes, and related documentation to the Master Software Development Library or the Software Development Libraries, and the subsequent release of the SI to integration and test, and system test.
- Provide status accounting of software work product changes submitted to the SDL or MSDL.
- Ensure that only approved changes are incorporated into the baselined software work products.
- Maintain a configuration audit system to ensure that records, which are provided to the MSDL

TABLE 1. Table 6.6 Software Development Library Levels and Controls-Example

Library Level	Library Name	Controlled By	SCCB	SCM
1	Software Work Area	Software Developer	Controls Promotion between levels	Controls Access to each Level
2	SCM Control, Integration and Test	Site SCM		
3	Software Build and Qualification Testing	SCM at Site or Subsystem		
4	Subsystem Qualification	Subsystem SCM		
5	System Verification and Validation	System SCM		

SCM = Software Configuration Management; SCCB = Software Configuration Control Board.

TABLE 2. Table 6.7 Comparison of Formal Software Baselines

Baseline Name	Baseline Contains	Where Baseline Exists
Functional	System requirements	Documents and databases
Allocated	Architectural Design	Set of documents (SRS, IRS, etc.)
Design	Detailed Design	Set of documents (SAD, SDD, etc.)
Development	Code under development	Source code
Product	Tested and approved system modules	Documentation and executable programs
Operational	Full system in use	Customer site(s)

or SDL, are consistent with documentation and software work product identification.

Three levels of configuration control are depicted in Figure 6.5 and five levels listed in example Table 6.6. Regardless of the number of CM control levels, overall responsibilities at each level should be defined in the SDP or SCMP including roles and procedures. In addition, the approach to CM related tasks should also be addressed (such as support to multiple baselines, a distributed development environment, data integrity and data restoration).

Scope Creep. There are many problems affecting (maybe infecting) the development of software but what makes it worse is that many Managers have a common misbelieve those are "not really problems"but simply characteristics of how software is developed. A lot of software projects fail or are canceled due to excessive time and cost overruns resulting from a non-existent, or poorly controlled, change management system.

For example, in order to be accommodating and a team player, a Manager may agree to what appears to be a "simple"change that seems like "a good idea,"even though the change is not a documented requirement. But, surprise, it turns out to be difficult to implement, and similar changes start to pile up, progress starts to slow down, and the entire project is thrown out of control. This is sometimes called scope creep. It is clear that changes are almost always needed, however, changes must be controlled through the change control process to avoid massive disruption and potential project cancelation or failure. (How many "little"projects have you ever started that turned out to be anything but little?)

6.3.5 Software Configuration Status Accounting

SCM must prepare and maintain records of the configuration status for all baselined software products and includes maintenance of the records required to support configuration auditing. Configuration status accounting data includes the current version of each baselined product, a record of changes to the software product since being placed under configuration control, and the recording and reporting of:

- When each baseline was created and when each SI completed the initial build placing the software or database under CM control.
- Descriptive information about each SI.
- Description and status of each Software Discrepancy Report (SDR). The status items could be: approved, disapproved, awaiting action, incorporated, or closed.
- Change status and traceability of changes to controlled software products.
- The status of the technical and administrative documentation associated with each product baseline and/or update to a product baseline.
- Closure and archive status.

6.3.6 Software Configuration Audits

SCM must *perform periodic configuration audits to verify that changes were made* in accordance with the

Corrective Action Process as described in your SDP or SCMP. SQA can witness and support these audits. Configuration Audits should be used to ensure that submitted software is accompanied by appropriate documentation and approvals, is correctly delivered and merged, and is correctly included in the software builds. The audits must ensure that each software entity incorporates only the approved changes scheduled for inclusion at the time of the audit. The degree of formality of the configuration audits may differ at the different levels of configuration control.

The Software Development Libraries (SDL) and the Master Software Development Library should be audited at least quarterly. A sampling of software releases must be checked against the *Software Version Description* (SVD) for correctness and completeness. You also must ensure that there is a SVD for each release.

Functional and Physical Configuration Audits. Software engineers may be requested to support the *Functional Configuration Audits (FCA), Physical Configuration Audits* (PCA) and in some cases a System Verification Review (SVR). Both software and system FCAs and PCAs may be conducted independently or concurrently. The SVR is often conducted concurrently with the System FCA. Your contract may require a Software FCA to be conducted as part of the System Qualification Test (SQT) or following the SQT. The purpose of a Software FCA is to demonstrate that each SI was successfully tested and complies with the software and interface requirements of its functional requirements and design documentation. To complete the Software FCA, Software and System Engineers must reach a technical understanding of the validity and degree of completeness of the *Software Test Reports* (STR) and the applicable software user documentation. If software FCAs are contractually required, they should be conducted on every SI in the system.

The Software FCA, discussed in Subsection 12.6.5, is a prerequisite to the software Physical Configuration Audit discussed in Subsection 12.7.1. The purpose of the Software PCA is to conduct a formal examination of the *as-built*, and *as-coded* SI, against its design documentation in order to establish the product baseline. The Software PCA includes a review of the *Software Product Specification* (SPS), the Interface Design Description (IDD), the *Software Version Description* and all the required operational and support documentation.

If there are differences between the physical configuration of the SI and the configuration used for the Software FCA, they must be identified at the Software PCA. Approved and outstanding changes against the SI must also be provided along with approved deviations and waivers to the requirements specifications. FCAs and PCAs may be conducted on a single SI, a related group of SIs, or incrementally such as blocks. Results of the Software PCA can become an entrance criterion for the System Verification Review discussed in Subsection 12.6.6.

bf 6.3.7 Packaging, Storage, Handling and Delivery

The SCM procedures for packaging, storage, handling, and delivery of software products must be provided in the SCMP for both the SDLs and the MSDL. Master copies of delivered software products must be maintained in the MSDL for the duration of the contract.

Packaging. The package for a software delivery, at a minimum, consists of the *Software Version Description* and the electronic media. The SwIPT responsible for the delivery prepares the SVD in collaboration with SCM. The SVD typically requires Engineering Review Board (ERB) and CCB approvals. SCM is responsible for providing the appropriate identification labels. SCM and SQA should perform the package content verification review, using a *Verification Checklist*. A hardcopy listing of the files should be attached to a signed hardcopy checklist.

Upon successful completion of the review and approval process, a formal transmittal contracts letter should be generated and submitted. Copies of the completed Verification Checklist and contracts letter must be maintained in the CM Library.

Storage. The storage requirement can be satisfied through the implementation of supporting library systems. There should be at least three basic components of the library system: a Software Library Management System; a Documentation Library; and Data Storage Backup:

- The Software Library Management System allows the software to be maintained in a central location, yet each host or client has access. A CASE tool can track the baseline changes, marking the transition throughout the activities of the Software Life Cycle. SCM must control the software libraries to provide a disciplined structure for progressive development, integration, test, and implementation of software within a controlled, well-defined environment.
- The Documentation Library is a repository for the most current approved and controlled documentation.
- The Data Storage Backup component provides daily incremental backups and system backups (performed daily or weekly) to ensure recovery from an uncontrollable situation.

Handling. A SCM CASE tool is ideally suited to administratively manage the handling of software through the version database directory structure. All elements in the database must be read-only-at all times. They

only become Read/Write when they are checked out for updating by an authorized user.

Delivery. Release packages for each increment must be delivered to the Program SCM organization. All deliveries must receive SwIPT approval prior to shipping. Software deliveries should be on removable media. Installation and checkout of the delivered products at customer-designated facilities should be performed if applicable.

If problems arise during the installation, checkout, or test, these problems must be documented in a *Software Discrepancy Report* and then resolved. SDRs are discussed in Section 6.4.

Delivery Preparation. SCM is responsible for accumulating the SIs for milestone deliveries. This responsibility includes overseeing the scheduling, storage, handling, and delivery of the project media. All SIs to be delivered must be examined by SQA for specification compliance, SOW compliance, open items unresolved, SDR closure, and test verification status. No delivery should leave the facility without proper authorization from the Program Director or designate. CM should retain records of all deliveries.

Software Version Description. An SVD document must also be prepared or updated for each release to provide a history of version changes, subsystem/element data, references to related documentation, and references to known problems. In preparation for delivery of the system to the customer, the CM organization must create the formal SVD document with detailed information on all software components, including COTS and reuse software, and their associated version numbers, plus special instructions necessary for system installation.

6.4 Managing the Software Corrective Action Process

The *Corrective Action Process* (CAP) is often called *change management.* Corrective action is triggered when performance deviates significantly from the plan, when defects that must be corrected are identified in the software work products, or when enhancements and improvements are proposed. A definition of "significant deviation" must be determined by mutual agreement between you and the customer. The opportunity to measure progress and identify issues that need Corrective Action can come from the reviews and evaluations, test results, and other quantitative management data. The CAP is described in the following three Subsections:

- Discrepancy Reports and Change Requests: Reporting problems and changes (6.4.1)
- Corrective Action System: Handling problems and issues for software work products (6.4.2)
- Configuration Control Boards: Hierarchical responsibilities of the change control boards (6.4.3)

6.4.1 Software Discrepancy Reports and Software Change Requests

To report problems or changes with baselined software products, *Software Discrepancy Reports and Software Change* Requests-or similar names-must be used as part of the Corrective Action Process. The SDR may also be called a "Software Deficiency Report" or a "Problem Report." The problem with calling everything a "problem" is that every issue is not a problem.

Lessons Learned. At one time I presented a report to program management that showed there were many Software Problem Reports being processed. I was asked to find out why we had so many problems. My investigation revealed that most of the problems were not really "problems." Many were requests for improvement, some were simple mistakes easily corrected (like a typo), and some were queries regarding observed discrepancies that needed to be reviewed to determine if they were potential problems. That was when we got rid of calling everything a "problem" and instituted the SCR and SDR.

The SCRs/SDRs are inputs to the Corrective Action Process discussed in Subsection 6.4.2. The following describes the basic differences between SCRs and SDRs:

- *Software Change Request:* An SCR is typically used to enhance or improve the software product or change commitments, plans or a baseline. It is inappropriate to classify recommended improvements or enhancements as a "problem." Also, if an anomaly is found early in the process, the change requested will prevent the problem from occurring later in the process when correcting it will be much more costly.
- *Software Discrepancy Report:* SDRs document unexpected error conditions or anomalies that occur and is deemed as an incorrect action (or reaction) of the software product.

Lessons Learned. At Lockheed Martin, when a change was made early in the development process that avoided a future problem, we called it a "save," and statistics were kept on the number of saves to show the cost saving advantages of early Peer Reviews. Most of the time, the cost savings of early Peer Reviews were impressive.

At the subsystem level, SCRs/SDRs are under the control of the Software Lead but may have to be passed to a higher-level board for approval. The Chief Software Engineer should be responsible for SCRs/SDRs at the program level. SCRs/SDRs must be used to report a known or suspected problem or discrepancy or change with software products under any level of configuration

control above the Developer of the product. The originator of the SCR/SDR should be responsible for completion of the *issue description* but not necessarily the person who fixes the issue.

Candidate items for inclusion into the CAP Tracking System are: project name, issue originator, item number, item name, software element or document affected, origination date, category, severity, issue description, analyst assigned to the problem, date assigned, date completed, analysis time, recommended solution, impacts, problem status, approval of solution, follow-up actions, name of corrector, correction date, version where corrected, correction time and description of the solution implemented.

6.4.2 Software Corrective Action Process

If you expect the unexpected and prepare for it, you can manage it when it happens!

A software *Corrective Action Process* must be implemented for your project in order to handle problems or issues detected in software work products under configuration control, and other problems or issues related to the contract. The CAP must use SCRs/SDRs as inputs to the process and the process should include mechanisms to ensure that:

- All detected problems and issues are promptly reported and entered into the CAP Tracking System, organized and checked for duplication.
- Corrective Actions are promptly initiated when the problems or issues are identified.
- Status is tracked and resolutions are achieved.
- Records of the problems and issues are maintained for the duration of the contract.
- Software problems are classified by category and severity.
- Analysis is performed to detect trends.
- Corrective Actions are evaluated to determine if changes are correctly implemented without introducing additional problems.

Details of the Corrective Action Process can vary from location to location. These details should be documented in each Developer's SDP Annex or in a Corrective Action Plan addendum to the SDP. The SCR/SDR process can also be detailed in the *Software Configuration Management Plan* and/or in lower level procedures.

The division of responsibilities for Corrective Action tasks between the Development Team and the customer's Program Office should be clear but is often confusing. A table should be included in the SDP to specifically clarify the division of responsibility. It can also be expanded to identify specific organizations performing each task.

Figure 6.6 is an example overview of a Corrective Action Process for Software Discrepancy Reports. As shown in Figure 6.6, once the SDR has been generated and logged in at the program level, the SDR is assigned to a Responsible Software Engineer for investigation. The investigator must recommend the Corrective Action needed and record the actions taken to either correct the problem or provide a workaround solution. When this is accomplished, the SDR is returned to the responsible Configuration Control Board for disposition.

Once a process problem is defined, it must be assigned a *priority and severity level.* Table 6.8 is an example of definitions for software problem severity levels. The status of problems must be reported and tracked. The CSwE, the SEPG, or both should perform trend analysis on process problems and report adverse trends to the appropriate level of management. Process issues are closed out when SQA verifies that the Corrective Action is in place and there exists objective

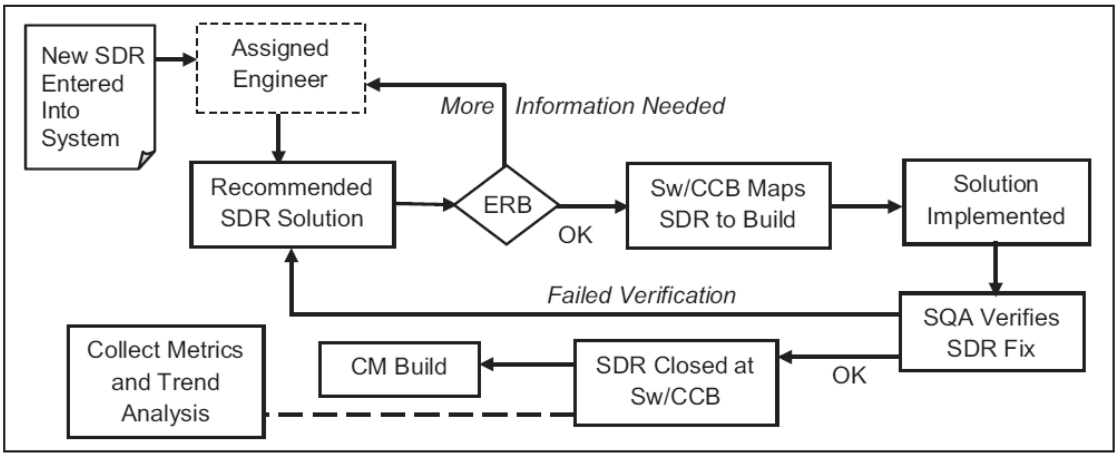

FIGURE 1. Figure 6.6 Corrective Action Process overview.