Software Configuration Management
(SCM)
Chapter_3

أ.م.د.عباس عبد العزيز عبد الحميد

كليـة العلـوم / قسم الحاسـوب

abbasabdulazeez@uomustansiriyah.edu.iq

3.8 Software Configuration Management

- The concept of configuration management (CM) was developed to manage changes in large systems.
- It handles the control of all product items and changes to those items.
- Software Configuration Management (SCM) is applied to software products.
- The product items include document, executable software, source code, hardware, and disks.
- SCM accrues two kinds of benefits to an organization:

  SCM ensures that development processes are traceable and systematic so that all changes are precisely managed.

  SCM enhances the quality of the delivered system and the productivity of the maintainers.
- The objectives of SCM are to:
  - Uniquely identify every version of every software at various points in time.
  - Retain past versions of documentations and software.
  - Provide a trail of audit for all modifications performed.
  - Throughout the software life-cycle, maintain the traceability and integrity of the system changes.

3.8 Software Configuration Management

Projects benefit from effective SCM as follows:

1. Confusion is reduced and order is established.
2. To maintain product integrity, the necessary activities are organized.
3. Correct product configurations are ensured.
4. Quality is ensured – and better quality software consumes less maintenance efforts.
5. Productivity is improved, because analysts and programmers know exactly where to find any piece of the software.
6. Liability is reduced by documenting the trail of actions.
7. Life cycle cost is reduced.
8. Conformance with requirements is enabled.
9. A reliable working environment is provided.
10. Compliance with standards is enhanced.
11. Accounting of status is enhanced.

3.8.2 SCM Spectrum of Functionality

- Estublier et al. classified the functionalities into three broad areas:
  - product,
  - tool, and
  - Process.

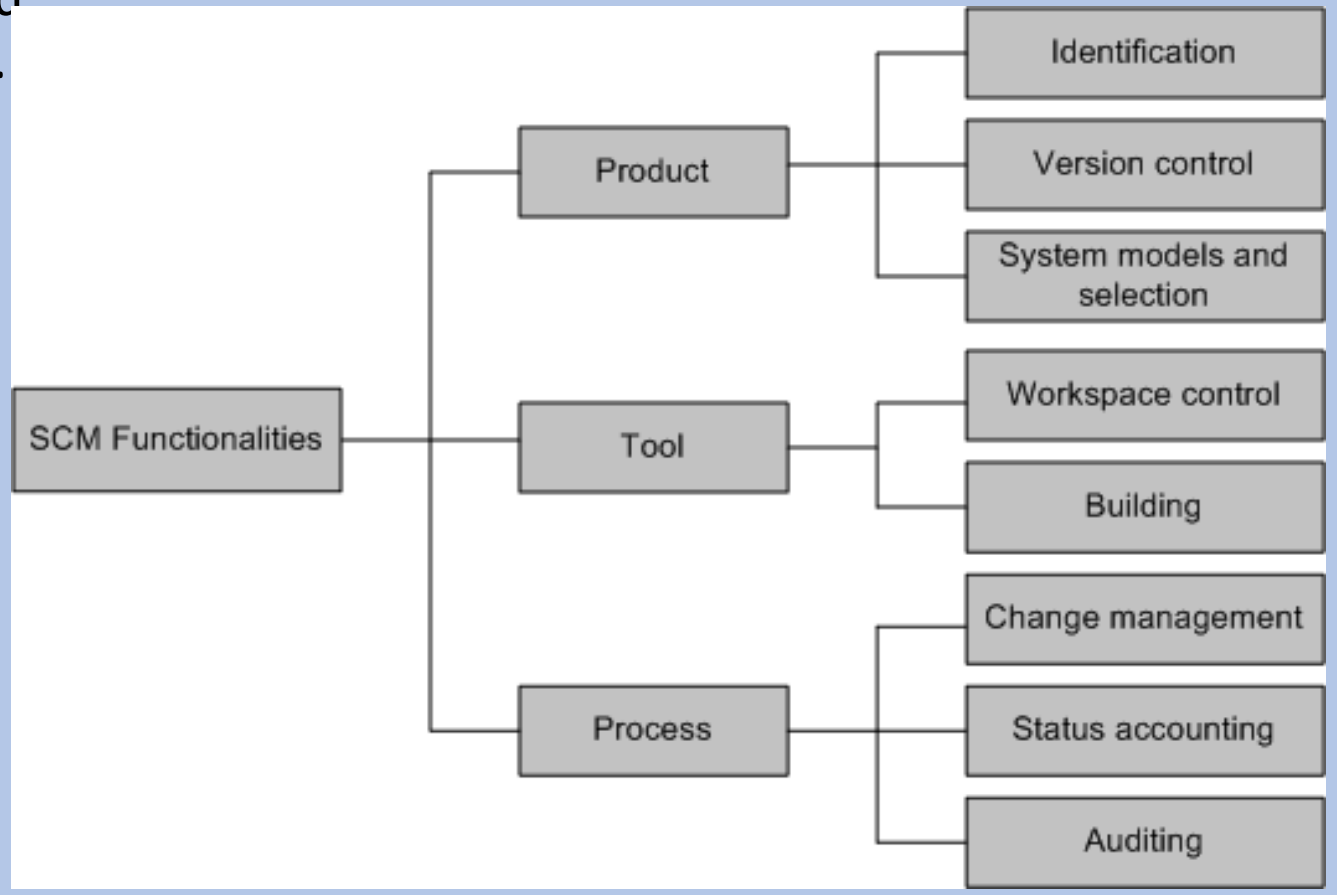

Figure 3.24 Technical dimension of SCM systems

3.8.2 SCM Spectrum of Functionality

Identification

- The items whose configurations need to be managed are identified in this function.
- The identified items include specification, design, documents, data, drawings, source code, executable code, test plan, test script, hardware components, and components of the software development environment, namely, compilers, debuggers, and emulators.
- Project plan and customer requirements should also be included.
- To accurately identify products, including their configuration and version levels, a schema of names and numbers is designed.
- Finally, for all configuration items and systems, a baseline configuration is established.

3.8.2 SCM Spectrum of Functionality

Version control

- To avoid confusion during the process of artifact evolution, a new identifier is assigned to the artifact every time the artifact is modified.
- One may be interested in recording a fact that a given artifact fixes a subset of defects found in an earlier release.
- The aforementioned kind of relation can be recorded by means of the version control (VC) functionality of SCM by:
  (i) interpreting software artifacts as configuration items, and
  (ii) identifying the relations, if there is any, among the configuration items.
- The basic version control idea is to have two separate files: master copies and working copies.
- The former is stored in a centralized repository.
- Software developers check out working copies from the repository, modify the working copies, and, finally, check in the working copies into the repository.
- Checking in a file means committing to the changes made to the working copies.

3.8.2 SCM Spectrum of Functionality

Version control

- Conflicts can arise if many software developers want to use the same version of a file.
- Conflicts can be resolved by means of two techniques: lock-modify-unlock and copy-modify-merge.
- Version control must support parallel development by allowing branching of versions.
- Consider the scenario: (i) an organization is currently developing the next version of their already released application; and (ii) a report about a major defect is received from the end users.
- Now the development group has the option to retrieve the released version and create a branch, as illustrated in Figure, to fix the defect.
- The figure illustrates how a file evolves with two branches, where the main path is called trunk. As erging with the trunk.

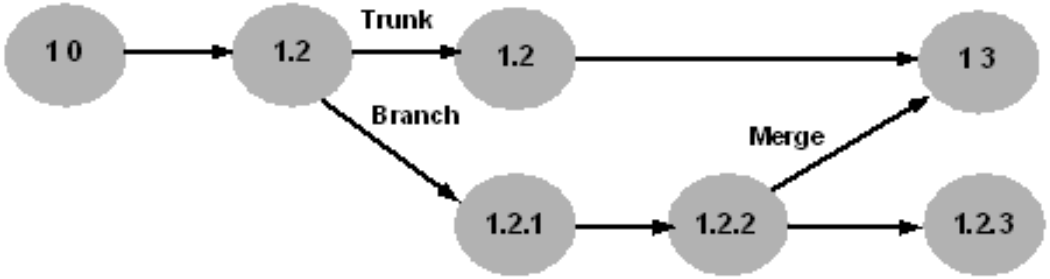

Figure 3.25 An evolution of a file with two branches

3.8.2 SCM Spectrum of Functionality

System models and selections

- It is neither efficient nor effective to manage a project file-by-file.
- There is a need to support aggregate artifacts so that maintenance personnel can enforce consistency in large projects by means of relationships among artifacts and attributes.
- Relationships among artifacts and attributes are captured by developing models which support the idea of software configurations.
- Intuitively, a configuration means an aggregate of versionable items.
- The general idea of configuration raises a need for enabling users to have selective access to parts and versions of such aggregated artifacts.
- By default, SCCS and RCS keep in the workspace the most recent version of the principal variant.
- Next, all artifacts that are exceptions to the default placement rule can be explicitly fetched by the user.

3.8.2 SCM Spectrum of Functionality

Workspace control

- An environment that enables the maintainer to make and test the changes in an isolated manner is called workspace.
- In an SCM system, software versions are stored in a repository that can not be directly modified. Rather, when a need to modify some files arises, the files are copied into a workspace.
- One can realize a workspace in two ways:

  (i) it can be as simple as the home directory of the programmer who wants to modify the files;

  (ii) it can be a complex mechanism such as an integrated development environment and a database.
- In general, three basic functions are performed in a workspace:
  - **Sandbox:** Checked out files are put in a workspace to be freely edited. In addition, it is not necessary to lock the original files in the repository.
  - **Building:** An SCM system generally stores the differences between successive versions to save space. Therefore, the workspace expands the deltas into full-fledged source files. In addition, the workspace stores the derived binaries.
  - **Isolation:** Every developer maintains at least one workspace. Therefore, the developer makes modifications to the source code, compiles the files, performs tests, and debugs code without impacting the works of other developers.

3.8.2 SCM Spectrum of Functionality

Building

- Efficiency is a key requirement of SCM systems so that developers can quickly build an executable file from the versioned source files.

- Second requirement of SCM systems is that it must enable the building of old versions of the system for recovery, testing, maintenance, or additional release purpose.

- Finally, SCM systems must support building of software.
- The build process and their products are assessed for quality assurance.
- Outputs of the build process become quality assurance records, and the records may be needed for future reference.

- The ***make*** application on the Unix operating system, originally developed by researchers at Bell Laboratories, is a classical example of a build process.
- Commercial SCM systems such as *ClearCase* continue to rely on variants of ***make.***

3.8.2 SCM Spectrum of Functionality

Change management

- SCM systems must:

  (i) enable users to understand the impact of modifications.

  (ii) enable users to identify the products to which a specific modification applies.

  (iii) provide maintenance personnel with tools for change management so that all activities from specifying requirements to coding can be traced.

- In the beginning, CRs were managed in paper form.
- However, these days CRs are saved in the SCM repository and are linked with the actual modifications, in addition to being automated.

3.8.2 SCM Spectrum of Functionality

Accounting

- The primary purpose of status accounting is to:
  (i) keep formal records of already existing configurations, and
  (ii) produce periodic reports about the status of the configurations.
  These records:
  - describe the product correctly,
  - are used to verify the configuration of the system for testing and delivery, and
  - maintain a history of change requests, including both the approved ones and the rejected ones.
- A history of change request includes the answers to the following questions:
  - Why are changes made?
  - When are the changes made?
  - Who makes the changes?
  - What changes are made?
- Status accounting is useful in communicating important details of the project and configuration items to the stakeholders of the project.

3.8.2 SCM Spectrum of Functionality

Auditing

- SCM systems need to provide the following features:

(i) roll back to earlier stable points.

(ii) identify which modifications were performed, why those modifications were performed, and who performed those modifications.

- By means of auditing, the organization maintains the integrity of the baselines and release configurations for all products.
- Two kinds of audits are performed before a software is released:

audit for functional configuration: determines whether or not the software satisfies the user requirement specification and the system requirement specification.

audit for physical configuration: It verifies if the reference and design documents accurately represent the software.

3.8.2 SCM Spectrum of Functionality

Auditing

- Overall, a **configuration audit** tries to find answers to the following:
  - To what extent are the requirements satisfied by the modified system?
  - Does the software release under consideration reflect the modification requests?
- The activities to perform a **configuration audit** are as follows:
  - Procedures and an audit schedule are defined.
  - The personnel to perform the audits are identified.
  - Established baselines are audited.
  - Audit reports are generated.

3.8.3 SCM Process

- Figure 3.26 shows the three major SCM implementation activities:
  - planning,
  - baseline development, and
  - configuration control.

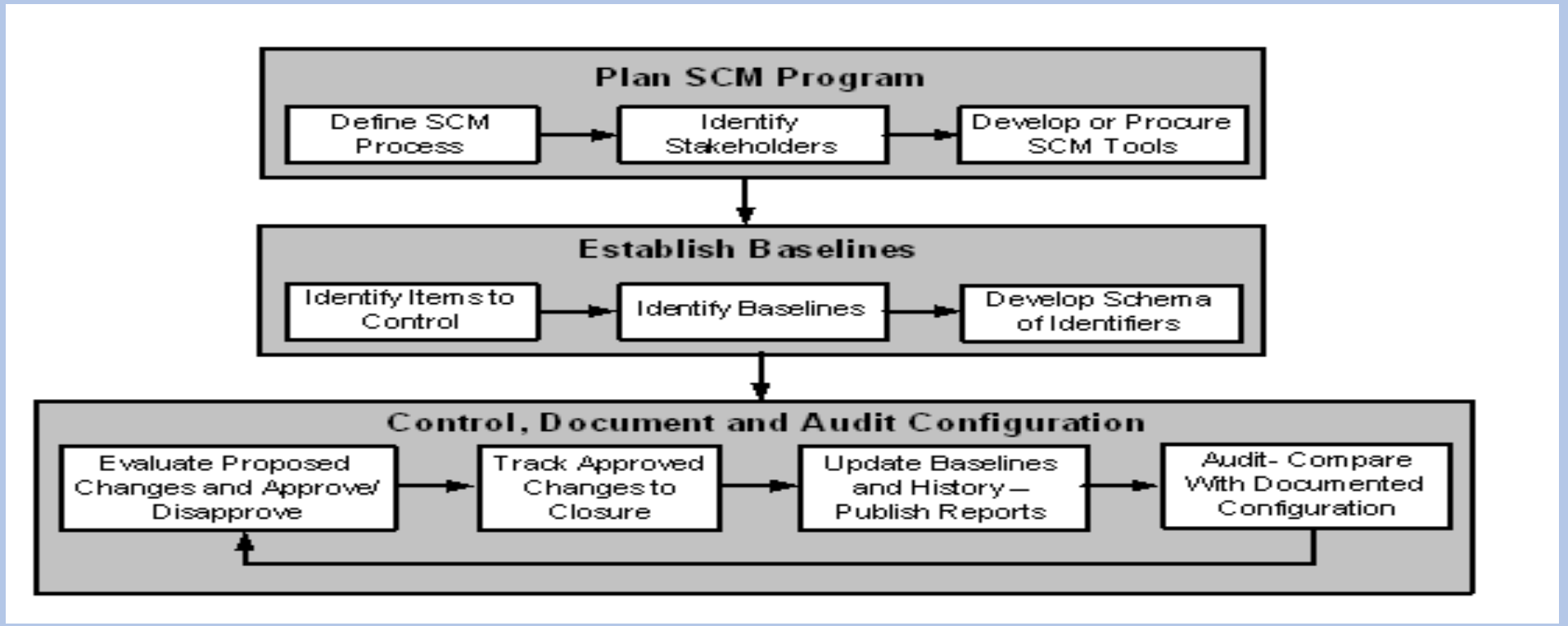

Figure 3.26 A process for implementing SCM

3.8.3 SCM Process

Planning

- Planning is begun with two activities:
  - (i) define the SCM process, and
  - (ii) establish procedures.
- A key step during planning is the identification of the stakeholders.
- The stakeholders in a configuration management are the maintainers, development engineers, sustaining test engineers, quality assurance auditors, users, and the management.
- The stakeholders are also known as configuration control board (CCB) members.
- Not all changes are reviewed by the board. Rather, small groups review and approve most of the changes.
- Therefore, those groups need to be identified in the planning phase.
- Various SCM tools are used to maintain configuration history and facilitate the SCM process flow.
- Examples of such tools are CVS (concurrent version system) and Clearcase.

3.8.3 SCM Process

Establishing baseline

- Once an SCM program plan is in place, the next step is identification of items (code, data, and documents) that are the subject of configuration control.
- After the configuration items identified, a software baseline library is established to make the set of configurable items publicly available.
- The library, called repository, is the heart of the SCM system.
- The repository has information about all the baselined items.
- The process of baseline (or re-baseline for a change) involve the following activities:
    1. Create a snapshot of the current version of the product and its configuration items, and allocate a configuration identifier to the entire configuration.
    2. Allocate version numbers to the configuration items and check in the configuration.
    3. Store the approved authority information as part of meta data in the repository.
    4. Broadcast all the above information to the stakeholders.
    5. To accurately identify the configuration version, design a schema of words, numbers, or letters for common types of configuration items. In addition, project requirements may dictate specific nomenclature.

3.8.3 SCM Process

Controlling, documenting and auditing

- After establishing a baseline, it is important to:

  (i) keep the actual and the documented configuration identical;

  (ii) ensure that the baseline complies with a project's configuration described in the requirements document.

- The aforementioned requirements of a baseline are realized by means of a four-step iterative process illustrated in Figure 3.26.

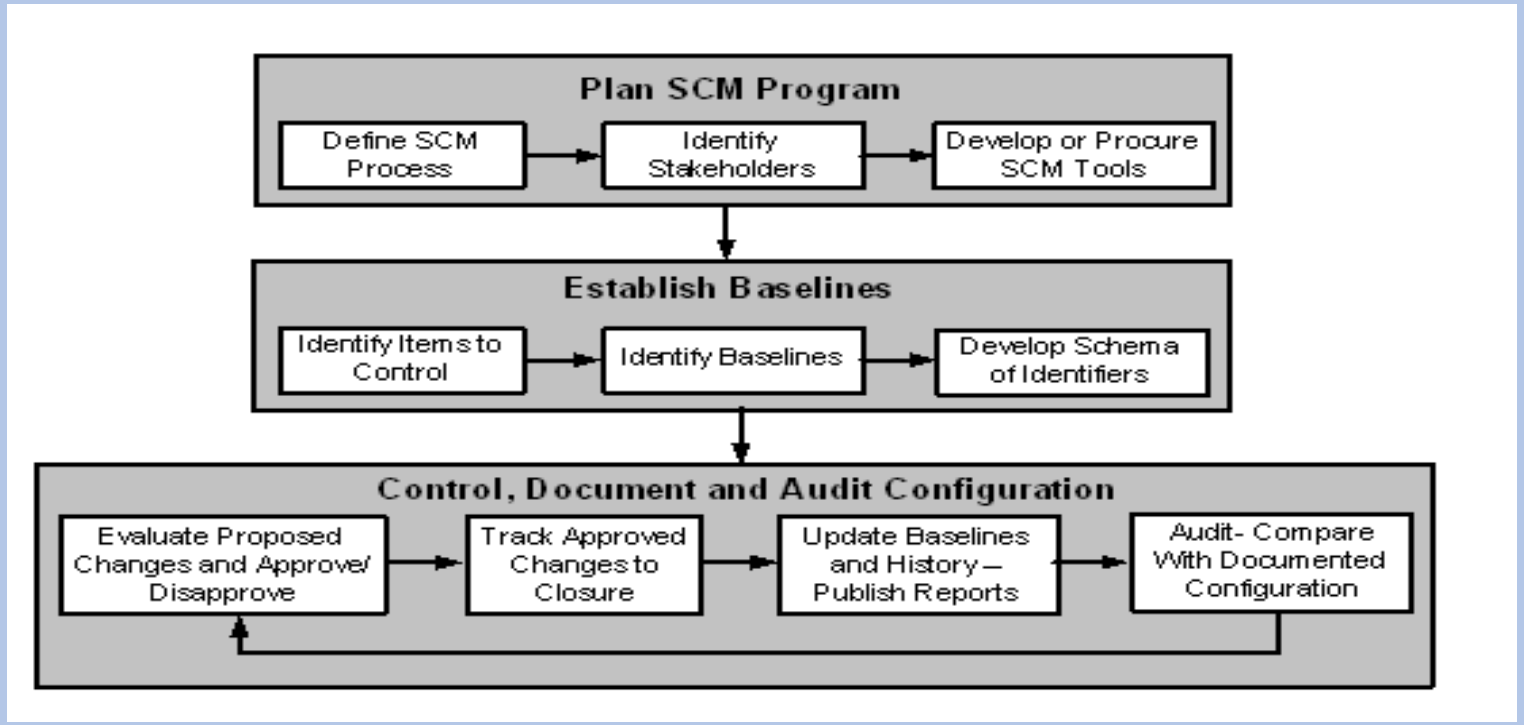

Figure 3.26 A process for implementing SCM

3.8.3 SCM Process

Controlling, documenting and auditing

- The stakeholders specified in the SCM plan review and evaluate all changes to the configuration.
- After their evaluations, both approvals and disapprovals are documented.
- Approved changes are tracked until they are verified .
- Next, the appropriate baseline is revised in conjunction with all relevant documents, and reports are generated.
- At regular intervals, records and products are audited to verify that:
  - There is acceptable matching between the documented configuration and the actual configuration.
  - The configuration conforms with the requirements of the project.
  - Documentations of all change activities are complete and up-to-date.
- The three steps in the cycle, namely, controlling, documenting, and auditing, are repeatedly executed throughout the lifetime of the project.

3.9 Change Request Workflow

- A change request (CR), also called a modification request (MR), is a vehicle for recording information about a system defect, requested enhancement, or quality improvement.
- Change requests are placed under the control of a change management system.
- Change management systems control changes by an automated system in the form of work-flow.
- The basic objective of change management is to uniquely identify, describe, and track the status of each requested change.
- The objectives of change management are as follows:
  - Provide a common method for communication among stakeholders.
  - Uniquely identify and track the status of each CR. This feature simplifies progress reporting and provides better control over changes.
  - Maintain a database about all changes to the system. This information can be used for monitoring and measuring metrics.

3.9 Change Request Workflow

- A change request describes the desires and needs of users which the system is expected to implement.
- While describing a CR, two factors need to be taken into account:
  - Correctness of CRs, and
  - Clear communication of CRs to the stakeholders.
- The results of interpreting a CR in different ways are as follows:
  - The team carrying out actual changes to the system and the team performing tests may develop contradicting views about the new system's quality.
  - The changed system may not meet the needs and desires of the end users.
- CRs need to be represented in an unambiguous manner, and made available in a centralized repository.
- Wide availability of CRs to all the stakeholders is likely to reveal differences in interpretations by different groups.

3.9 Change Request Workflow

- The life-cycle of a CR has been illustrated in Figure 3.27, by means of a state-transition diagram.
- Each state represents a distinct stage in the life-cycle of a CR.

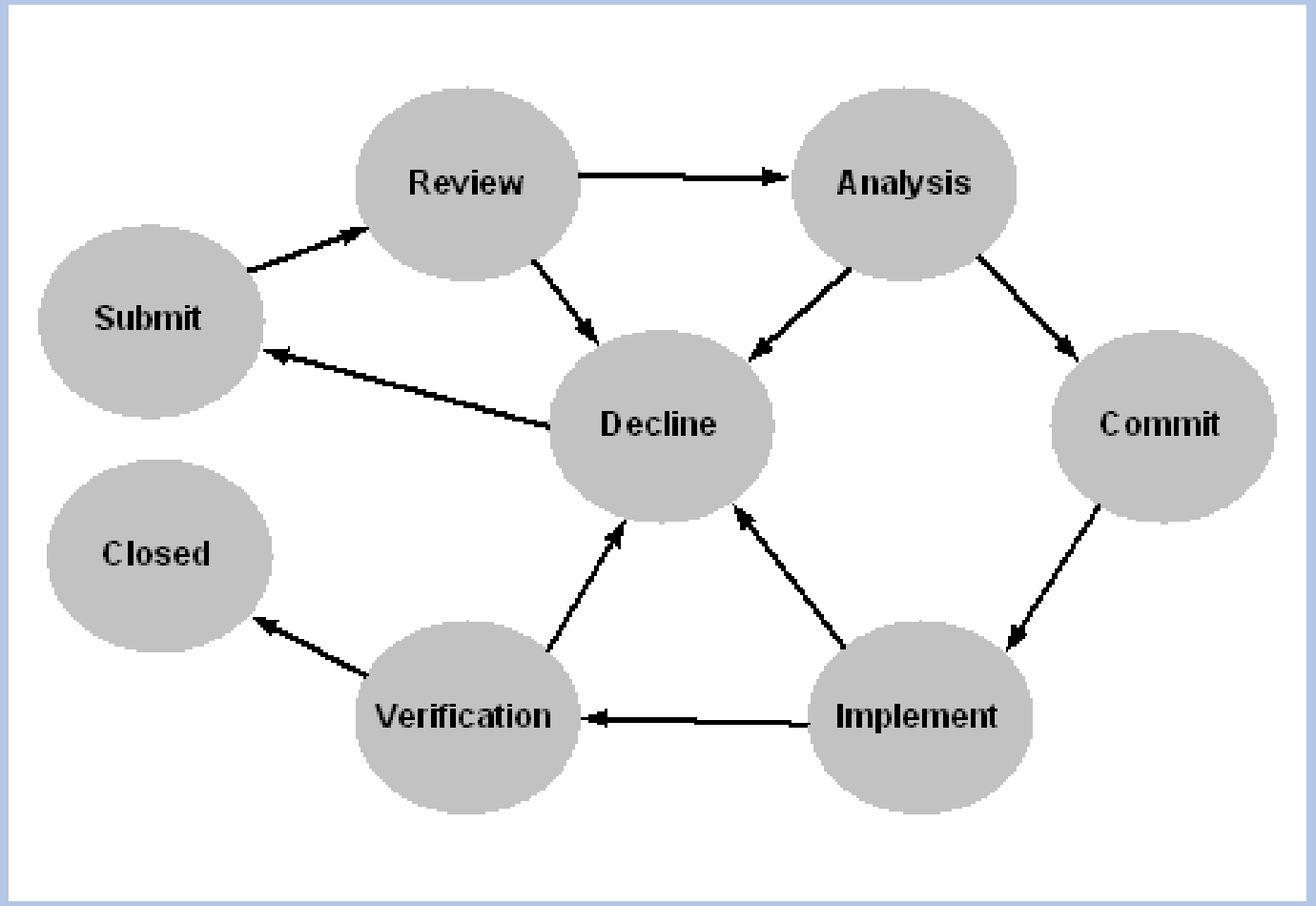

Figure 3.27 State transition diagram of a CR