Chapter 5

Software Configuration Management

Software Configuration Management

- ❖Change is **inevitable** when computer software is built
- ❖Change increases the level of **confusion** when you and other members of a software team are working on a project.
- ❖Confusion arises when changes are not analyzed before they are made, recorded before they are implemented
- ❖The **art of coordinating** software development to minimize confusion is called configuration management .

Software Configuration Management

- ❖**Configuration management is the art of identifying, organizing and controlling modifications to the software being built by a programming team**
- ❖The goal is to **maximize productivity** by minimizing mistakes.
- ❖Software Configuration Management (SCM) is an **umbrella activity** that is applied throughout the software process because change can be occur at any time.

SCM activities are Developed to

1. **Identify change**
2. **Control Change**
3. **Ensure** that change is being properly implemented
4. **Report** the change to others who may have an interest

Difference between SCM and Software Support

- Support is a set of software engineering activities that occur after software has been delivered to the customers and put into operation
- Software configuration management is a set of tracking and control activities that are initiated when a software engineering project begins and terminates only when the software is taken out of operation .

software process

1. Programs (both source level and executable format)
2. Documents
3. Data

- The items that comprise all information produced as part of the software process are collectively called a **software configuration**
- As the software process progresses, the number of software configuration items (SCIs) grows rapidly
- A hierarchy of SCIs is created as each SCI creates new SCIS.
- There will be little confusion if any SCI are involved but there is always a factor called Change that generates a lot of confusion

Software Process

- Change May occur at **anytime**, for any reason
- **First Law of System Engineering states** “No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle”

Fundamental Source of Change

- **New business or market conditions** dictate changes in product requirements or business rules
- New **customer** need demands
- Business **growth/downsizing** causes the change
- **Budgetary or scheduling** constraints cause a redefinition of the system or product.
- **Software Configuration Management is a set of activities that have been developed to manage change through the life cycle of Computer Software.**
- SCM can be viewed as a **software Quality Assurance Activity** that is applied throughout the software process

Baseline

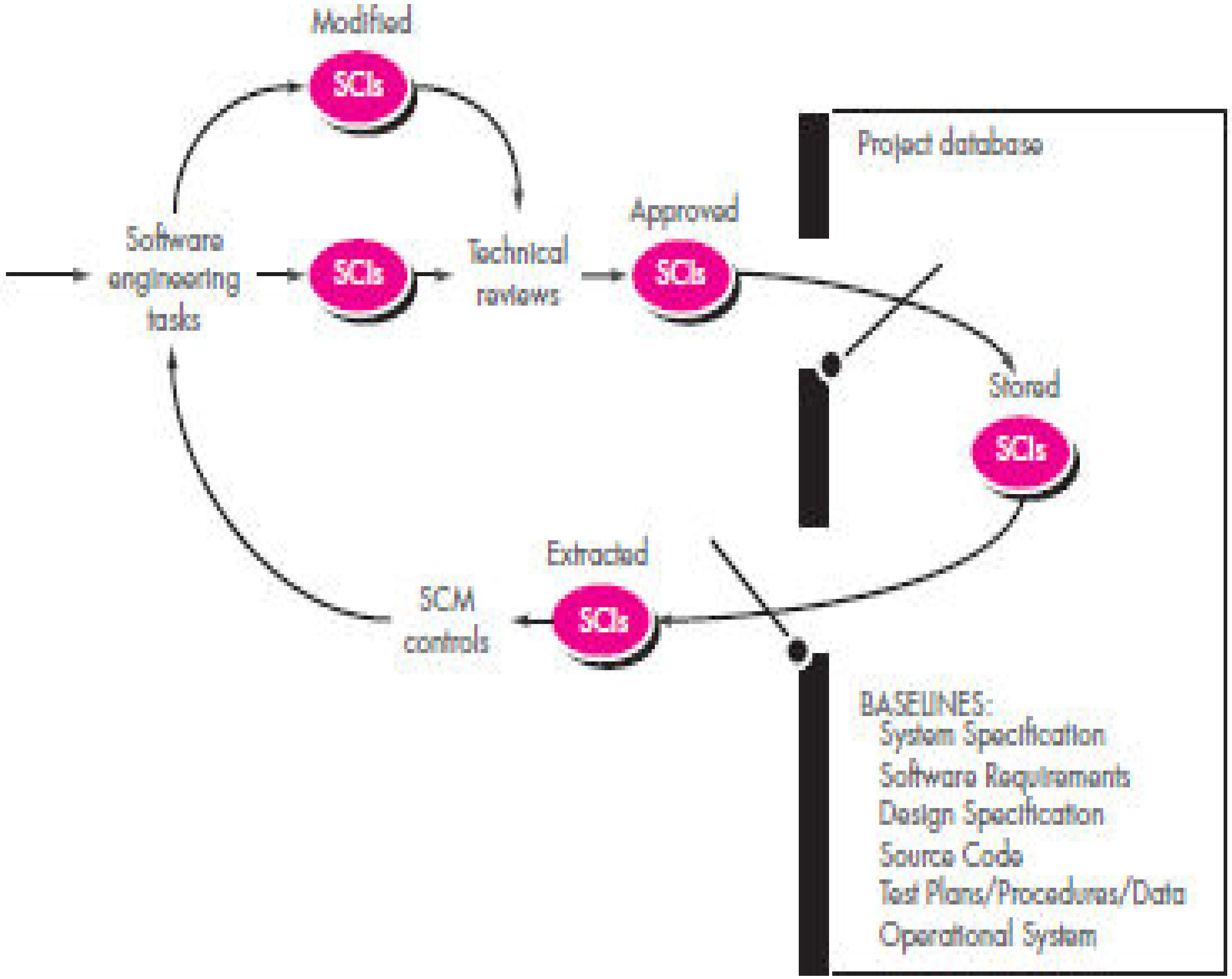
- Change is a fact in software Development
  - ***Customers want to modify the requirements***
  - ***Developers want to modify the technical approach***
  - ***Mangers want to modify the project strategy***
- A baseline is a software configuration management concept that helps you to **control change**
- Before a software configuration item becomes a baseline, changes maybe made **quickly and informally.**

Baseline

- Once a baseline is established, **changes can be made**, but a specific formal procedure must be applied to evaluate in every each change.

FIGURE 22.1

Baselined SCIs and the project database



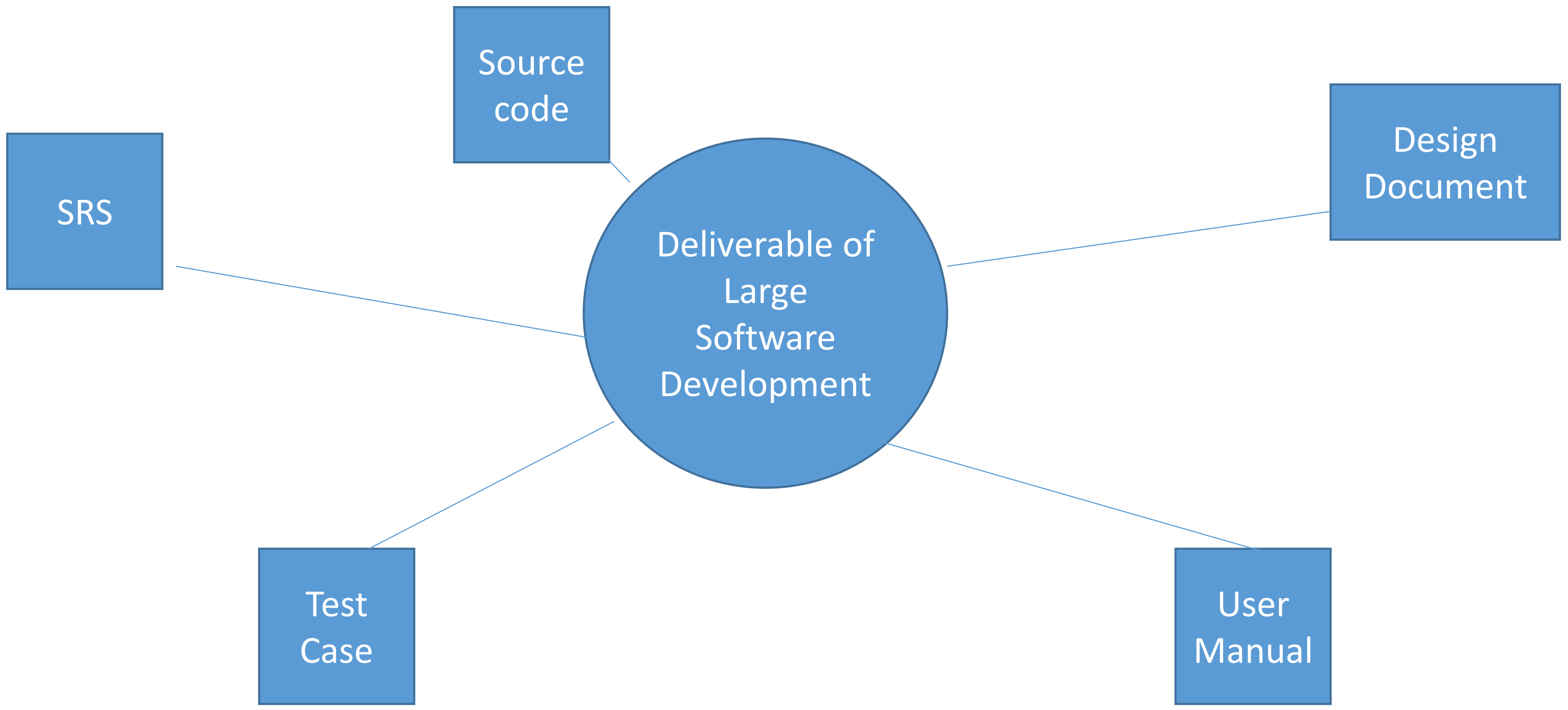
Baseline

- Software engineering task produce one or more SCIs
- After SCIs are reviewed and approved they are placed in a project database (Known as software repository)
- When a member of a software engineering team wants to make a modifications to a baseline SCI, it is copied from the project database into the Engineering's private work space
- However, this extracted SCI can be modified only if SCM controls are followed.

Software Configuration Items

- Software configuration item as **information** that is **created** as part of the software engineering process.
- SCI could be considered to be a single section of a large specification or one test case in large suite of tests
- SCI is all or part of a work product (e.g a document , an entire suite of test cases, or a named program component)

Software Configuration Items



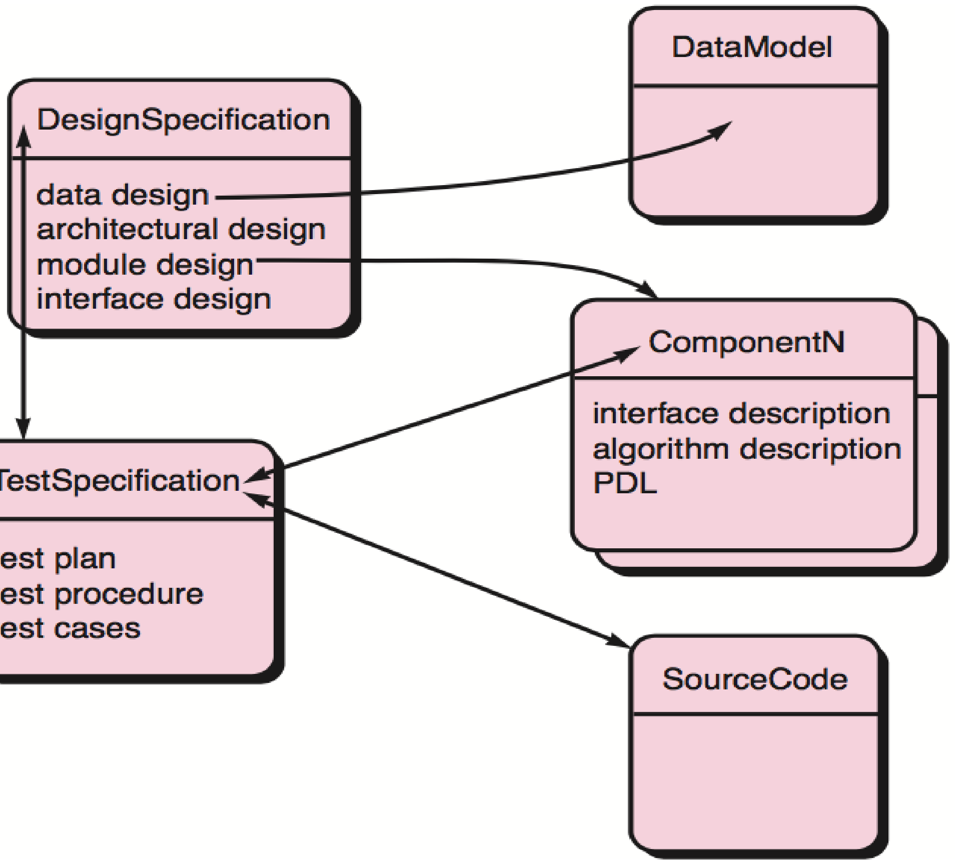
Software Configuration Items

- SCIS are organized to form ***configuration objects*** that may be cataloged in the project db with a single name
- A **configuration object** has a **name**, **attributes**, and is connected to the other objects by relationships
- **Design Specification**, **Data Model Component** , **Source Cod**e, and Test Specification are each identified separately

Configurations Objects

FIGURE 22.2

Configuration objects



Software Configuration Items

- Each of the objects is related to the others as shown by the arrows
- A curved arrow indicates a compositional relation
- That is, **Data Model** and **ComponentN** are part of the object **Design Specification**
- A double-headed straight arrow indicates an interrelationship.
- If a change were made to the **Source Code** object, the interrelationships enable you to determine what other objects (and SCIs) might be affected.

SCM Repository

- In the early days of software engineering, software configuration items were maintained as paper documents.

This approach was problematic for many reasons:

(1) finding a configuration item when it was needed was often difficult,

(2) determining which items were changed, when and by whom was often challenging,

(3) constructing a new version of an existing program was time consuming

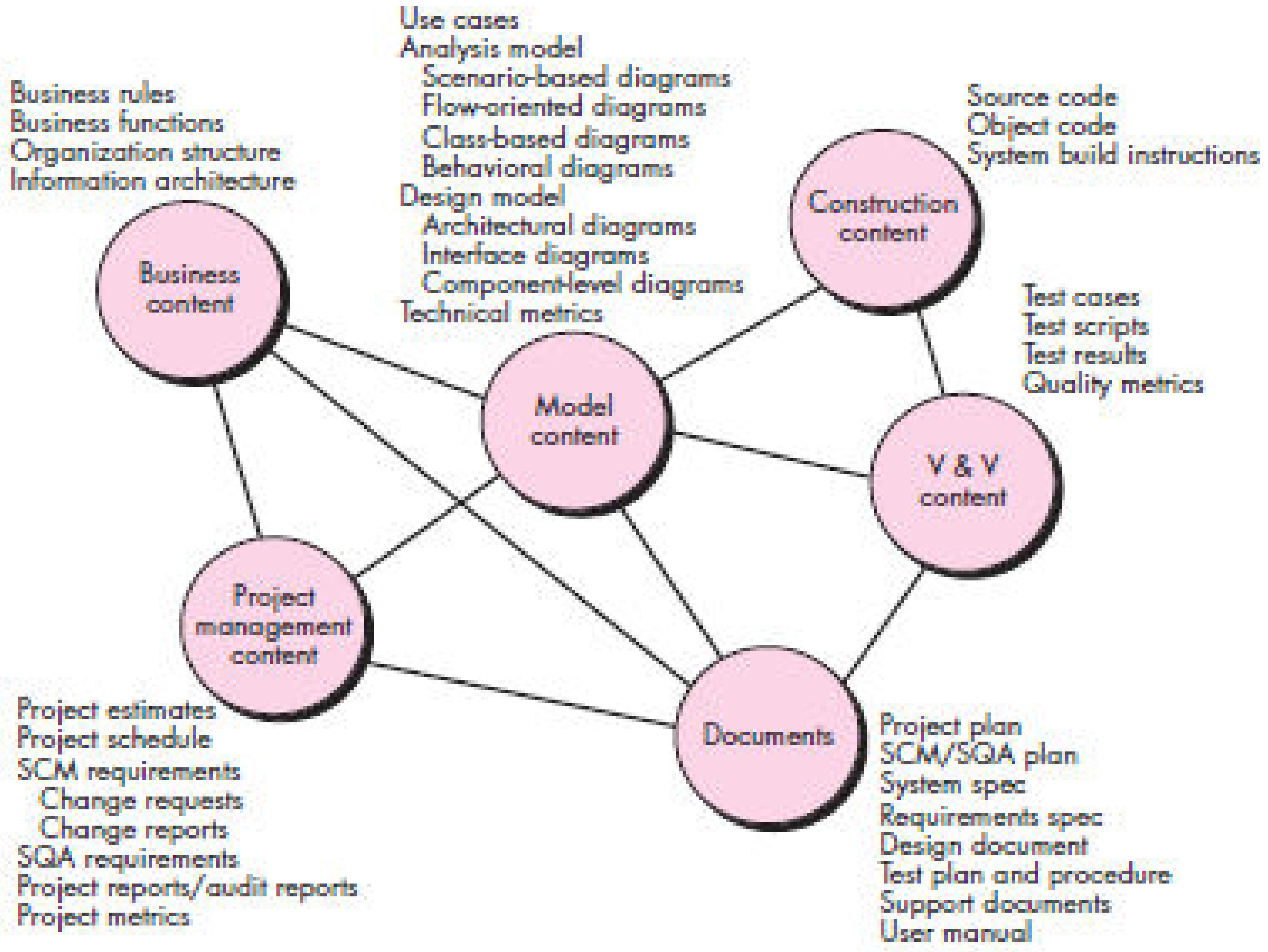
and error prone

(4) describing detailed or complex relationships between configuration items was virtually impossible.

SCM Repository

- Today, SCIs are maintained in a project **database or repository.**
- The SCM repository is the set of mechanisms and data structures that allow a software team to manage change in an effective manner.
- It provides the obvious functions of a modern database management system by **ensuring data** integration functions of a modern database management system by ensuring data integrity, sharing, and integration.

Content of the repository



The SCM Process

- The software configuration management process defines a series of tasks that have four primary objectives:

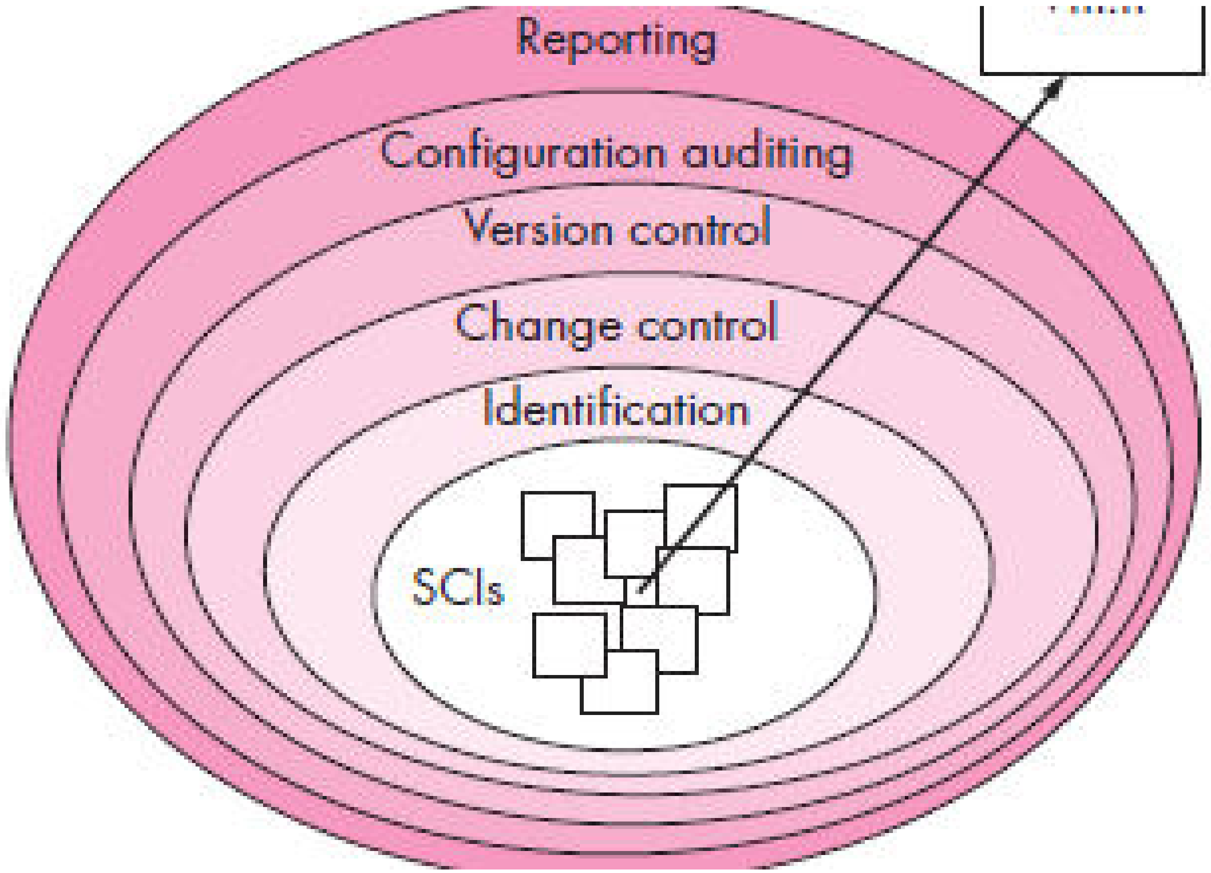
  (1) to **identify all items** that collectively define the software configuration,

  (2) to **manage changes** to one or more of these items,

  (3) to **facilitate** the construction of different versions of an application, and

  (4) to **ensure** that software quality is maintained as the configuration evolves over time.

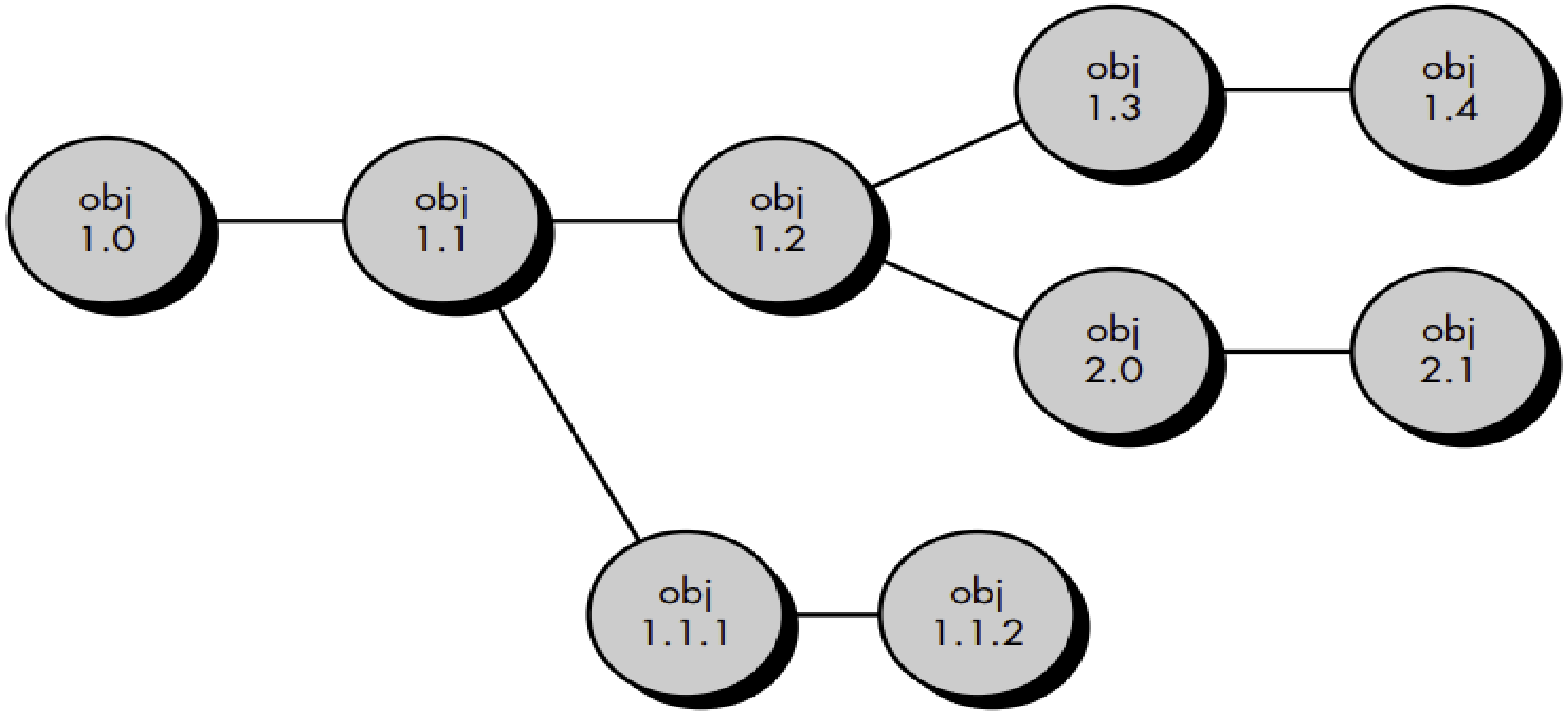
Layers in Software Configuration Management



Identification of Objects

- To control and manage software configuration items, each should be **separately named** and then **organized using** an object-oriented approach.
- Different types of objects are identified - **Basic Objects**, **aggregated objects**
- Each object has a set of distinct features that identify it uniquely: **a name**, a **descriptio**n, a list of **resources**, and a “**realization.”**

Identification of Objects



Identification of Objects

- The identification scheme for software objects must recognize that objects evolve throughout the software process.
- Before an object is baselined, it may change many times, and even after a baseline has been established, changes may be quite frequent.
- It is possible to create an evolution graph for any object.

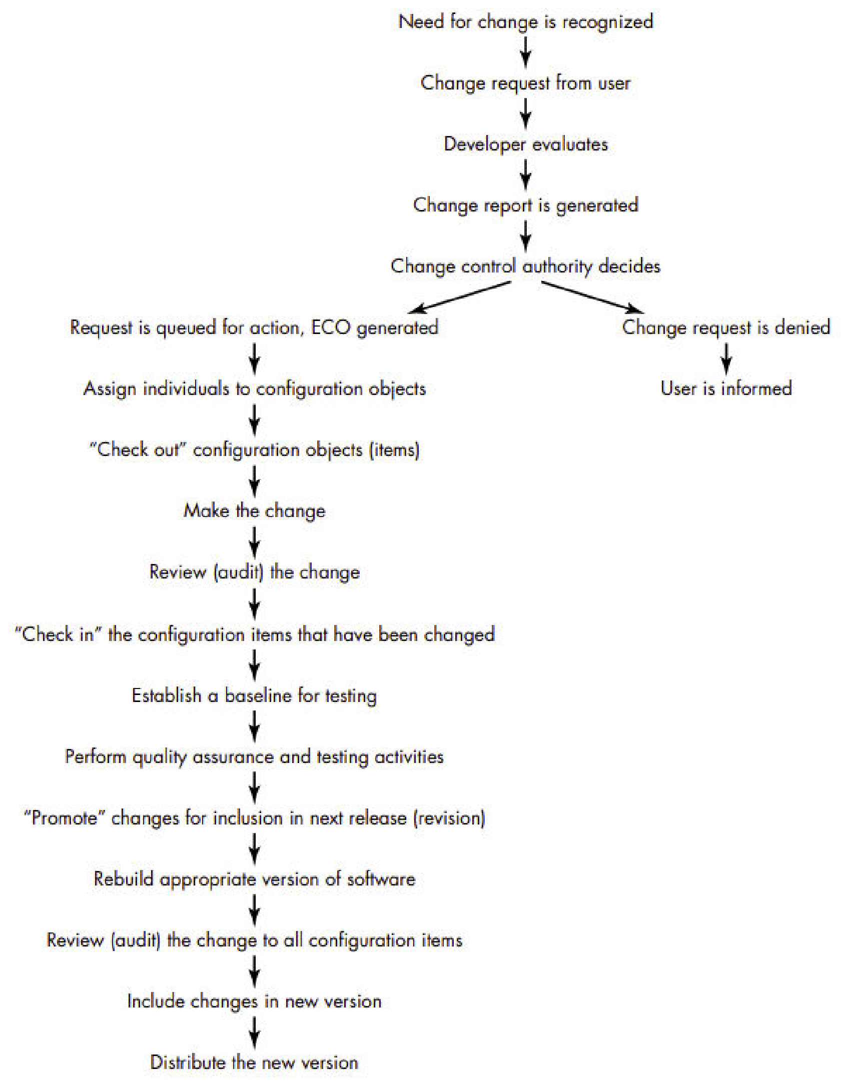
Identification of Objects

- The evolution graph describes the change history of an object, as illustrated .
- Configuration object 1.0 undergoes revision and becomes object 1.1.
- Minor corrections and changes result in versions 1.1.1 and 1.1.2, which is followed by a major update that is object 1.2.
- The evolution of object 1.0 continues through 1.3 and 1.4, but at the same time, a major modification to the object results in a new evolutionary path, version 2.0.
- Both versions are currently supported

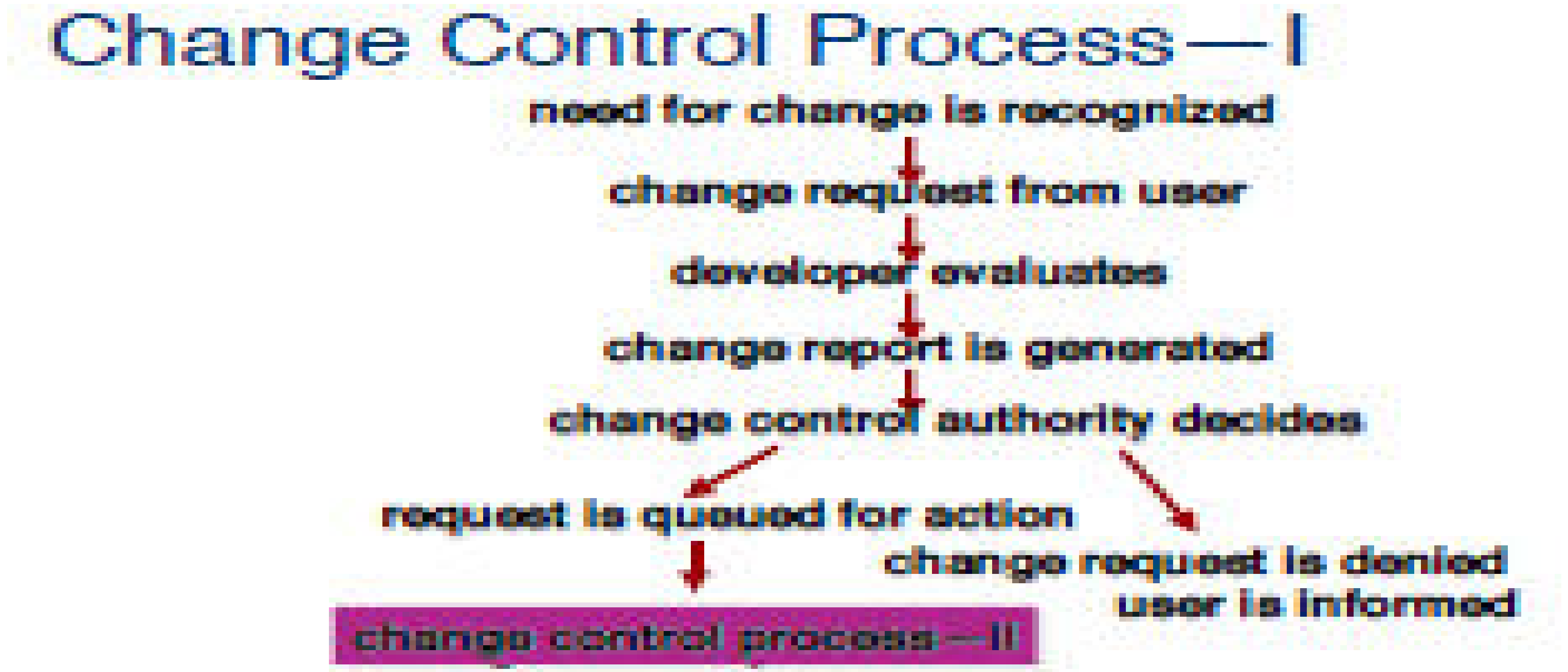
Change Control

- A ***change request*** is submitted and evaluated to assess **technical merit**, **potential side effects**, overall impact on other **configuration objects** and **system functions**, and the projected cost of the change.
- The results of the evaluation are presented as a *change report,* which is used by a *change control authority* (CCA)—a person or group that makes a final decision on the status and priority of the change.
- **An *engineering change order* (ECO)** is generated for each approved change.
- The ECO describes the change to be made, the constraints that must be respected, and the criteria for review and audit.
- The object to be changed can be placed in a directory that is controlled solely by the software engineer making the change.
- A version control system updates the original file once the change has been made

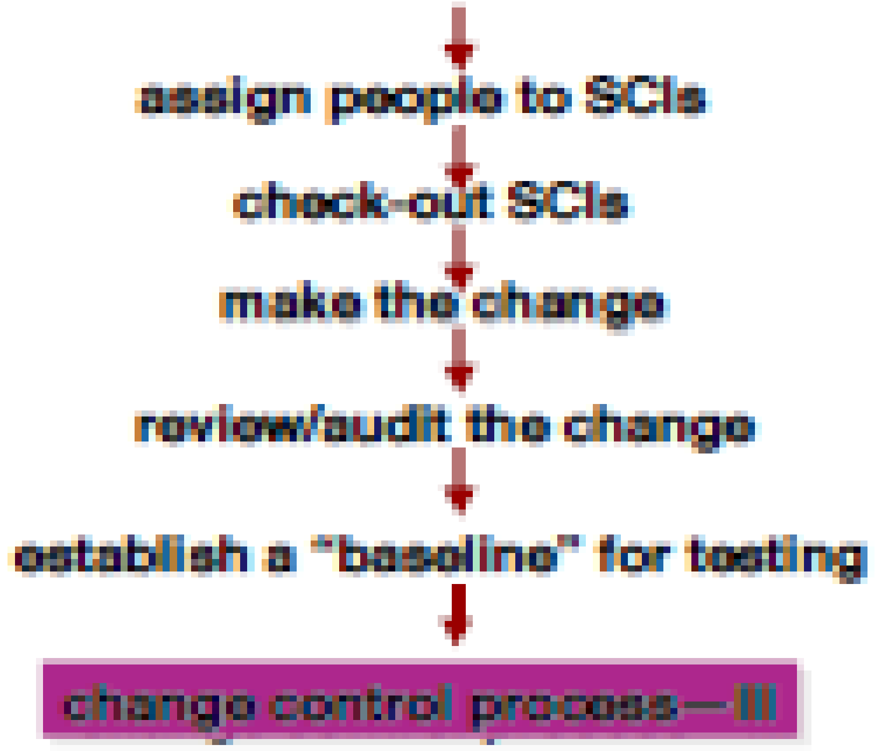
ss


Need for change is recognized
Change request from user
Developer evaluates
Change report is generated
Change control authority decides
Request is queued for action, ECO generated
Change request is denied
User is informed
Assign individuals to configuration objects
"Check out" configuration objects (items)
Make the change
Review (audit) the change
"Check in" the configuration items that have been changed
Establish a baseline for testing
Perform quality assurance and testing activities
"Promote" changes for inclusion in next release (revision)
Rebuild appropriate version of software
Review (audit) the change to all configuration items
Include changes in new version
Distribute the new version

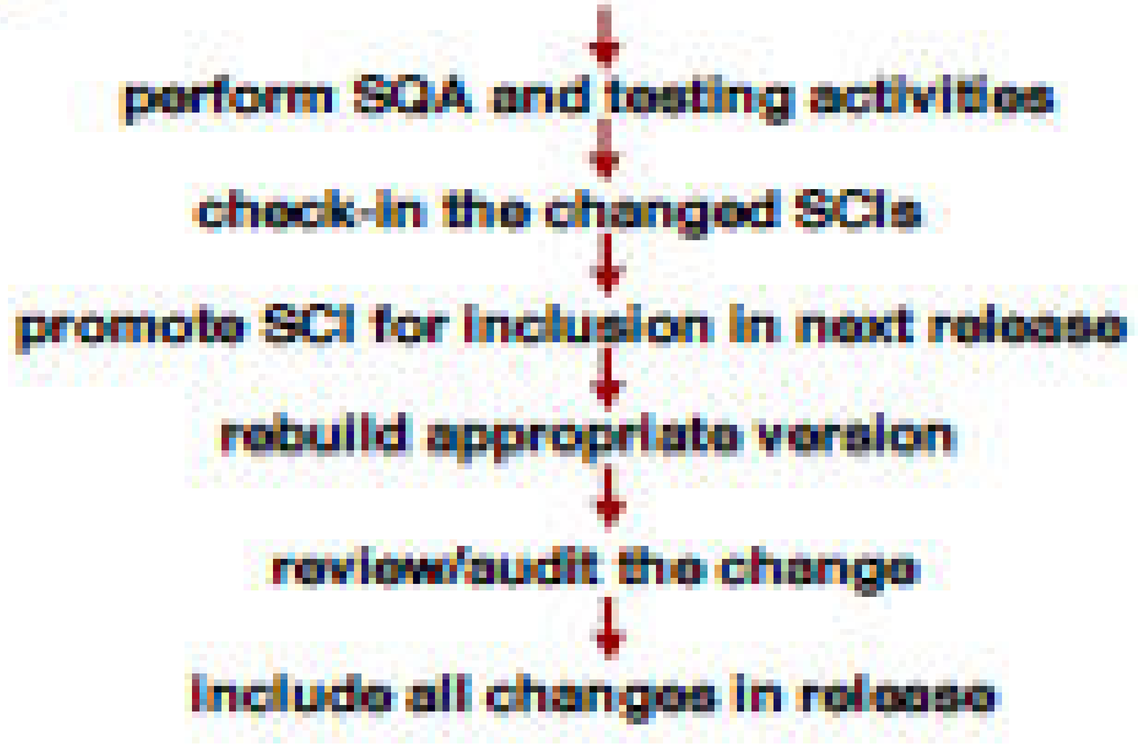
Change Control


Change Control Process—I
need for change is recognized
change request from user
developer evaluates
change report is generated
change control authority decides
request is queued for action
change request is denied
user is informed
change control process—II

Change Control Process-II



Change Control Process-III


perform SQA and testing activities
check-in the changed SCIs
promote SCI for inclusion in next release
rebuild appropriate version
review/audit the change
include all changes in release

Version Control

- Version Control Combines procedures and tools to manage different version of configuration objects that are created during the software process.
- A version control system implements or is directly integrated with four major capabilities:
    1. A project database that stores all relevant configuration objects,
    2. A version management capability that stores all version of configuration object,
    3. A make facility that enables the software engineer to collect all relevant configuration objects, and
    4. Construct a specific version of the software.

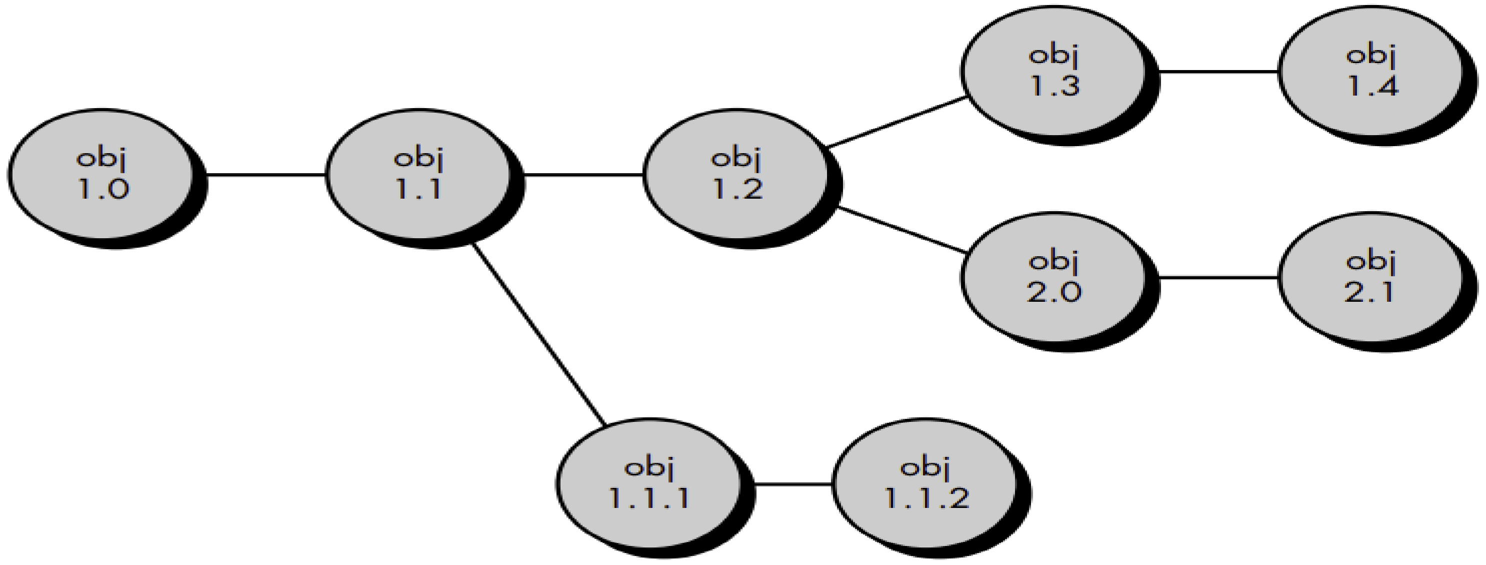
Version Control

- A number of version control systems establish a set – a collection of all changes (to some baseline configuration) that are required to create a specific version of the software.
- “Changes set” captures all changes to all files in the configuration along with reason for changes and details of who made the changes and when.
- A number of named change set can be identified for an application or system.
- This enables a software engineer to construct a version of the software by specifying the changes set (by name) that must be applied to the baseline configuration.

Version Control

- Version control combines procedures and tools to manage different versions of configuration objects that are created during the software process.
- One representation of the different versions of a system is the evolution graph.
- Each node on the graph is an aggregate object, that is, a complete version of the software.

Version Control



Version Control

- Each version of the software is a collection of SCIs (source code, documents, data), and each version may be composed of different variants.
- To illustrate this concept, composed of entities 1, 2, 3, 4, and 5
- Entity 4 is used only when the software is implemented using color displays.
- Entity 5 is implemented when monochrome displays are available. Therefore, two variants of the version can be defined:

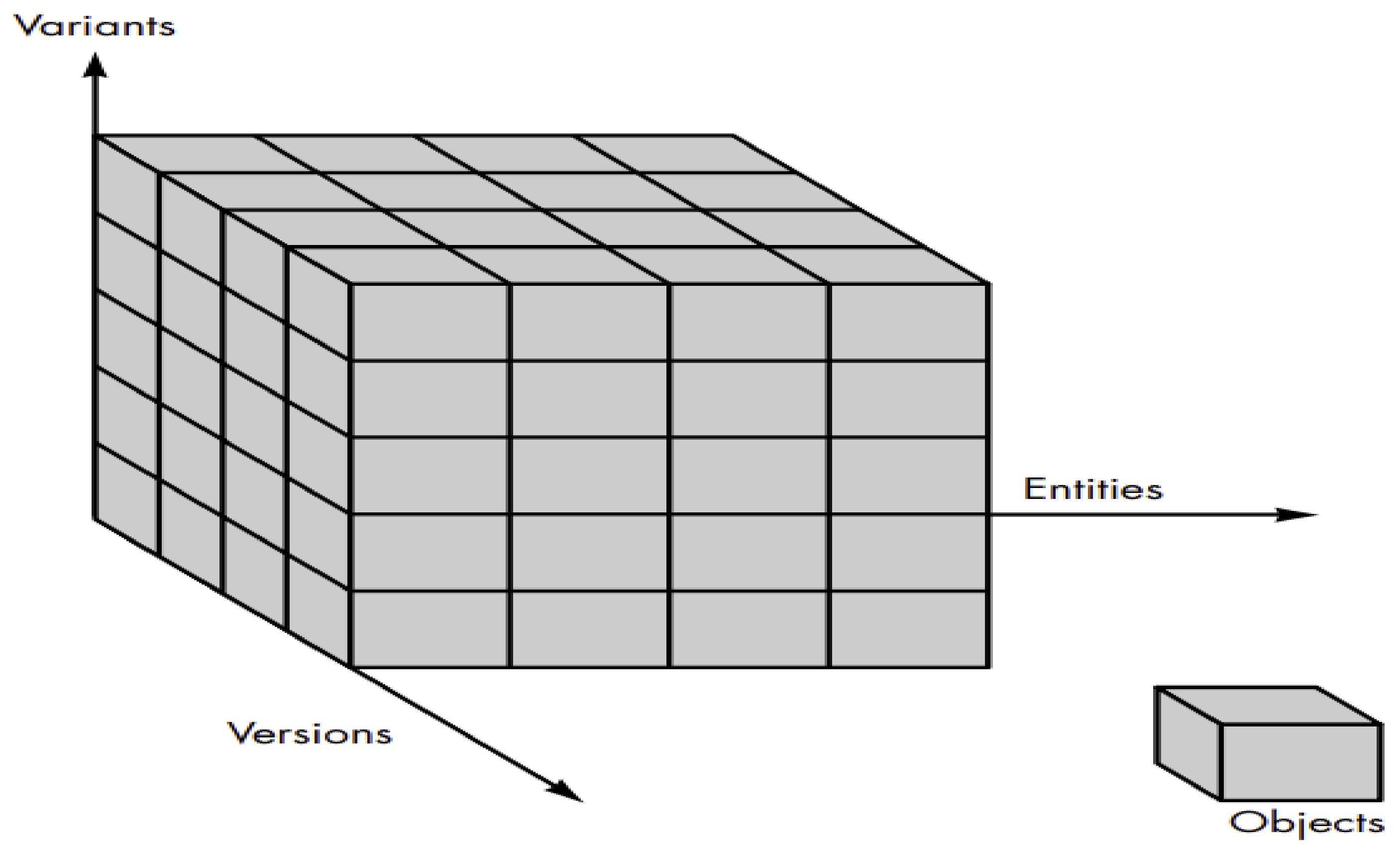
(1) entities 1, 2, 3, and 4;

(2) entities 1, 2, 3, and 5.

Version Control

- Another way to conceptualize the relationship between entities, variants and versions (revisions) is to represent them as an object pool .
- Referring to F the relationship between configuration objects and entities, variants and versions can be represented in a three-dimensional space.
- An entity is composed of a collection of objects at the same revision level.
- A variant is a different collection of objects at the same revision level and therefore coexists in parallel with other variants.
- A new version is defined when major changes are made to one or more objects.

Version Control



Configuration Audit

- How can we ensure that the change has been properly implemented?
  - (1) formal technical reviews
  - (2) the software configuration audit.
- The formal technical review focuses on the technical correctness of the configuration object that has been modified.
- The reviewers assess the SCI to determine consistency with other SCIs, **omissions, or potential side effects**.
- A formal technical review should be conducted for all but the most **trivial changes.**

Configuration Audit

- *Software configuration audit* complements the technical review.
- The audit asks and answers the following questions:
- Has the change specified in the ECO been made? Have any additional modifications been incorporated?
- Has a technical review been conducted to assess technical correctness?
- Has the software process been followed and have software engineering standards been properly applied?
- Has the change been “highlighted” in the SCI?
- Have SCM procedures for noting the change, recording it, and reporting it been followed?
- Have all related SCIs been properly updated?

Configuration Audit

- In some cases, the audit questions are asked as part of a formal technical review.
- However, when SCM is a formal activity, the SCM audit is conducted separately by the quality assurance group.

Configuration status reporting

- *Configuration status reporting also* called ***status accounting*** is an SCM task that answers the following questions:

(1) What happened?

(2) Who did it?

(3) When did it happen?

(4) What else will be affected?

SCM Features

- ❖Versioning.
- ❖Dependency tracking and change management.
- ❖Requirements tracing.
- ❖Configuration management.
- ❖Audit trails.