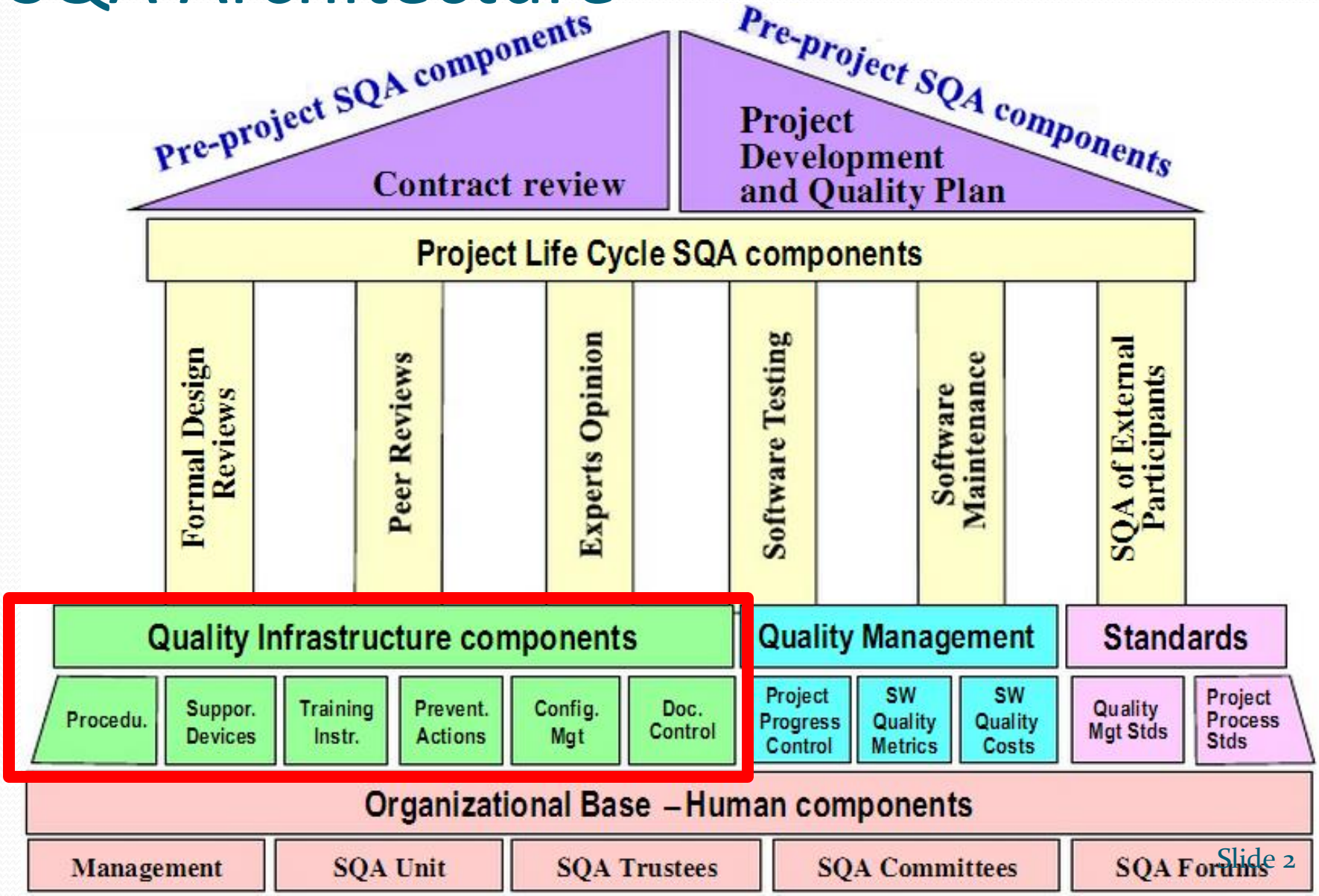
Software quality infrastructure components

1 Overview	2 Life cycle components	3 Infrastructure components	4 Management components	5 Standards and Organizing
6 Static tesing	7 Dynamic testing	8 Test management	9 Tools	

SQA Architecture



References

- Galin (2004). *Software Quality Assurance from theory to implementation.* Pearson Education Limited
- Ian Sommerville (2011). *Software engineering*. Ninth Edition. Addison-Wesley

Learning objectives

- Explain the **procedures**, **work instructions**, **templates**, **checklists** of software quality assurance
- Explain the main objectives of **training** and **certification** and list the main components of a certification program
- Explain the difference between **corrective** and **preventive actions**
- Describe the software **configuration management** activities
- Describe the tasks involved in establishment and maintenance of a **controlled documents list**

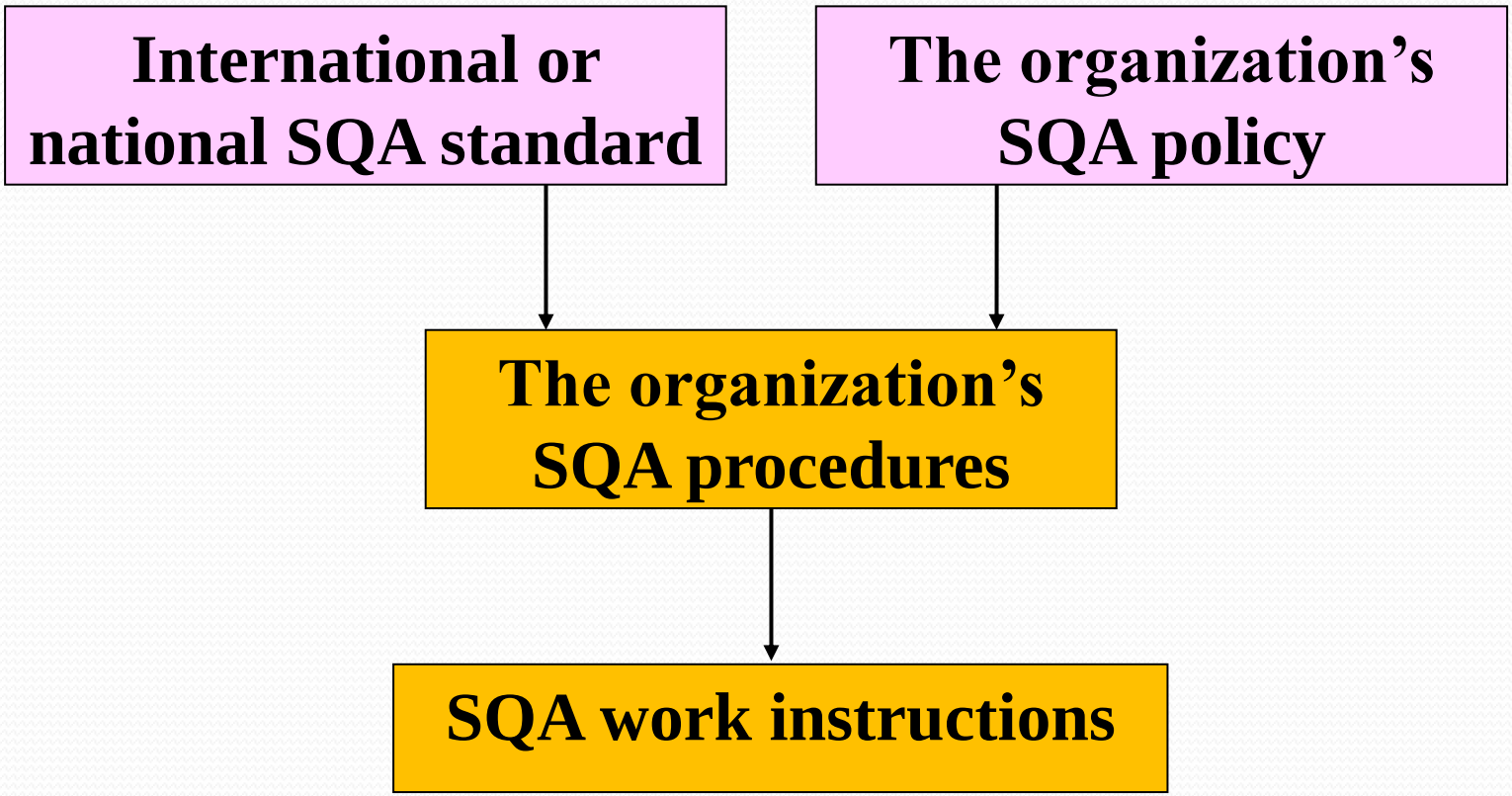
Contents

- Procedures and work instructions
- Quality support devices
- Staff SQA training and certification activities
- Corrective and preventive actions
- Configuration management
- Documentation control

Procedure and Work instruction

- **Procedure**
  - the detailed activities or processes to be performed according to a given method for the purpose of accomplishing a task
  - five W's
    - What activities have to be performed?
    - How should each activity be performed?
    - When should the activity be performed?
    - Where should the activity be performed?
    - Who should perform the activity?
  - tend to be universal within the organization
- **Work instruction**
  - detailed directions for the use of methods
  - specific to specific project team, customer, or other relevant party

Conceptual hierarchy of procedures and work instructions



Example: Fixed table of contents for procedures

1. Introduction *
2. Purpose
3. Terms and abbreviations *
4. Applicable documents
5. Method
6. Quality records and documentation
7. Reporting and follow-up *
8. Responsibility for implementation *
9. List of appendices *

See an example in Appendix 14A

The need for Procedures and Work instructions

- Performance of tasks, processes or activities in **the most effective** and **efficient way** without deviating from quality requirements
- Effective and efficient **communication** between development and maintenance teams that reduces the misunderstandings which lead to software errors
- **Simplified coordination** between tasks and activities performed by various teams that means fewer errors

Procedures and work instruction: preparation, implementation, updating

- The activities involved in maintaining an organization's procedures manual
  - preparation of new procedures
  - implementation of new or revised procedures
  - updating procedures

Contents

- Procedures and work instructions
- **Quality support devices**
- Staff SQA training and certification activities
- Corrective and preventive actions
- Configuration management
- Documentation control

Quality support devices

- Templates
- Checklists

Quality support devices Templates

- A **format** (especially table of contents) created by units or organizations, to be applied when compiling a report or some other type of document

- Example:
  - Software Test Plan (STP), Software Test Description (STD), Software Test Report (STR), Software Requirement Specification (SRS), etc.

Quality support devices
The software test plan (STP) – example

1 Scope of the tests
- 1.1 The software package to be tested (name, version and revision)
- 1.2 The documents that provide the basis for the planned tests (name and version for each document)

2 Testing environment
- 2.1 Testing sites
- 2.2 Required hardware and firmware configuration
- 2.3 Participating organizations
- 2.4 Manpower requirements
- 2.5 Preparation and training required of the test team

3 Test details (for each test)
- 3.1 Test identification
- 3.2 Test objective
- …

Quality support devices
The contribution of templates

- To development team:
  - **facilitates** the process of preparing documents
  - documents prepared are more **complete**
  - provides for **easier** integration of new team members
  - **facilitates** review of documents
- To software maintenance team:
  - enables **easier location** of the information

Quality support devices Checklists

- **List of items** specially constructed for each type of document to be completed prior to performing an activity
- Example

Goldenbug Ltd

Checklist for requirement specification report

Project name: _____

The reviewed document: _____ Version: _____

Item no.	Subject	Yes	No	N.A.*	Comments
1	**The document**				
1.1	Prepared according to configuration management requirements				
1.2	Structure conforms to the relevant template				
1.3	Reviewed document is complete				
1.4	Proper references to former documents, standards, etc.				
2	**Specifying the requirements**				
2.1	Required functions were properly defined and clearly and fully phrased				
2.2	Designed inputs conform with required outputs				
2.3	Software requirement specifications conform with product requirements				

Quality support devices
The contribution of checklists

- To development teams:
  - helps developers carrying out **self-checks** of documents or software code prior completion
  - assists developers in their **preparations** for tasks
- To review teams:
  - **assures completeness** of document reviews by review team members
  - **facilitates** improves efficiency of review sessions

The organizational framework of templates, checklists

- Preparation of new templates, checklists
- Application of templates, checklists
- Updating templates, checklists

Preparation of new templates, checklists

- Include senior staff, the department's chief software engineer and SQA unit members
- The most common information sources used in preparing a template, checklists:
  - already used in the organization
  - examples found in professional publications
  - used by similar organizations

Application of templates, checklists

- Several fundamental decisions are involved in the implementation of new or updated templates, checklists:
  - what channels should be used for advertising the templates, checklists?
  - which templates, checklists will be compulsory and how can their application be enforced?

Updating templates, checklists

- Sources for updating templates, checklists:
  - user proposals and suggestions
  - changes in the organization's area of activity
  - analysis of failures as well as success
  - other organization's experience
  - SQA team initiatives

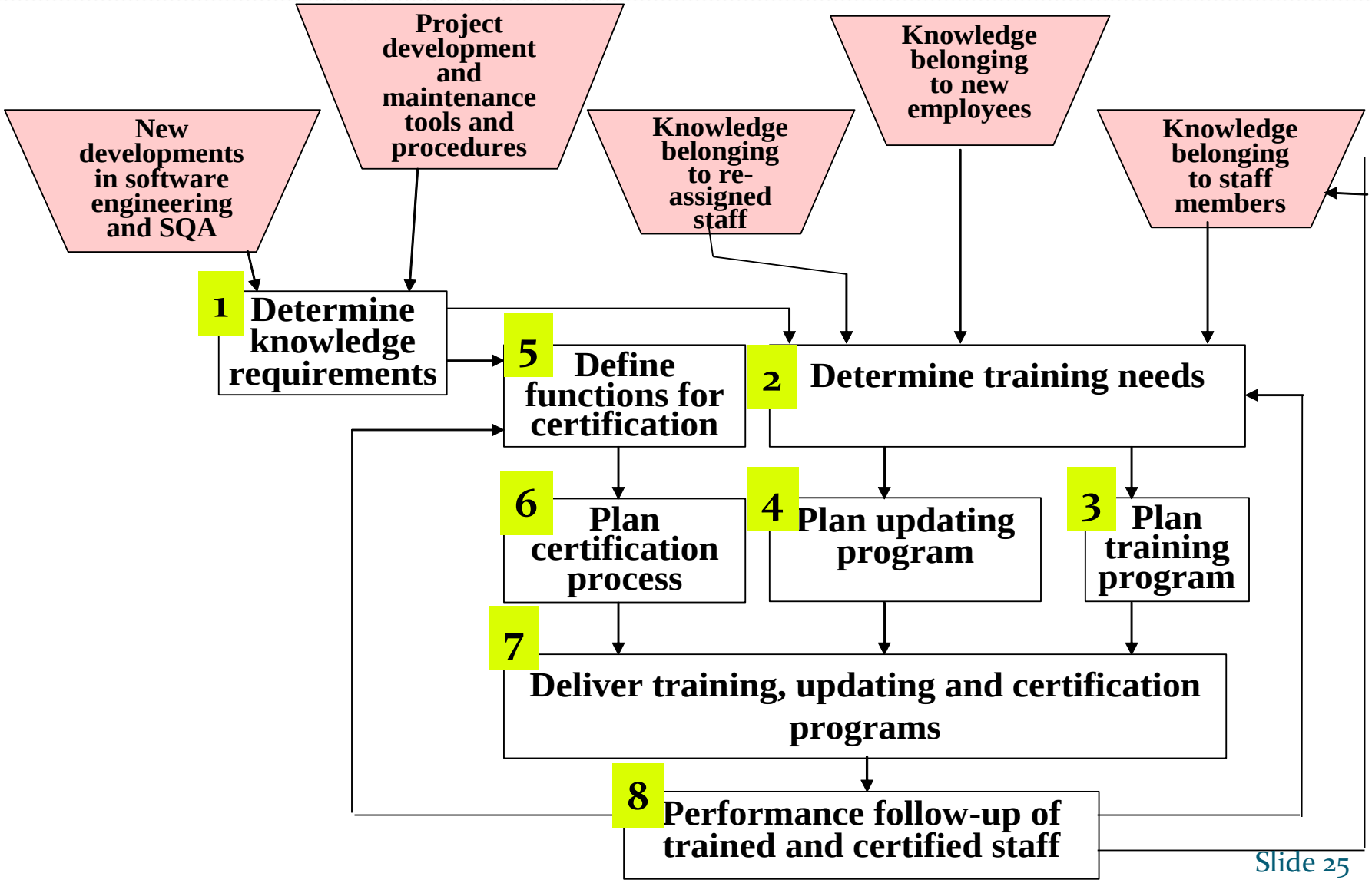
Contents

- Procedures and work instructions
- Quality support devices
- **Staff SQA training and certification activities**
- Corrective and preventive actions
- Configuration management
- Documentation control

Training and Certification

- Objectives
  - to **develop** the knowledge and skills needed by new employees
  - to **update** the knowledge and skills of veteran employees
  - to **transmit** knowledge of SQA procedures
  - to **assure** that candidates for key positions are adequately qualified
  - to **assure** conformity to the organization's standards for software products (documents and code)

The training and certification process



The training and certification process
1. Determining professional knowledge requirements

- Staff members still need additional “local” knowledge and skills
- This can be grouped into two categories
  - knowledge and skills of software engineering topics
  - knowledge of SQA topics

The training and certification process
2. Determining training and updating needs

- The type of training is adapted to the needs of three distinct groups of staff:
  - **training**: for new employees, according to their designated assignment
  - **retraining**: for employees assigned to new positions or receiving new assignments
  - **professional updating**: for staff members as demanded by their position

The training and certification process
3-4. Planning training and updating programs

- For software engineering:
  - On-the-job or e-learning

- For SQA (organized periodically)
  - training for new employees, or updating the existing employees
  - typical SQA updating program: once a year or once every six months

The training and certification process
5. Defining positions requiring certification

- Examples
  - software development team leader, programming team leader, software testing team leader, software maintenance technician and internal quality auditor
- A **certification committee** defines the list of positions that require certification and whether the certification will be effective permanently or for a limited period
  - should be revised periodically
  - varies by firm or organization

The training and certification process
6. Planning the certification processes

- Typical certification requirements:
  - professional education: academic or technical degrees
  - internal training courses
  - professional experience in the organization (may be partially or completely in other organizations)
  - assessment of achievements and ability
  - evaluation by the candidate's direct superior
  - demonstration of knowledge and skills by means of a test or a project

The training and certification process
7. Delivery of training and certification programs

- How training and updating are carried out?
  - short lectures and demonstrations (lasting only half a day)
  - lengthy courses (over several weeks or months)
  - may be conducted in-house, by the organization's training unit, or externally, by vocational or academic institutions

The training and certification process
8. Follow-up activities

- Follow-up is meant to provide the information necessary to initiate revisions of the training and certification programs based on **performance data**
- Sources for performance data
  - questionnaires completed by trainees, their superiors and others
  - analysis of outstanding achievements as well as failures
  - specialized review of software products (documents and code) produced by certified and trained employees

Contents

- Procedures and work instructions
- Quality support devices
- Staff SQA training and certification activities
- **Corrective and preventive actions (CAPA)**
- Configuration management
- Documentation control

Corrective and preventive actions

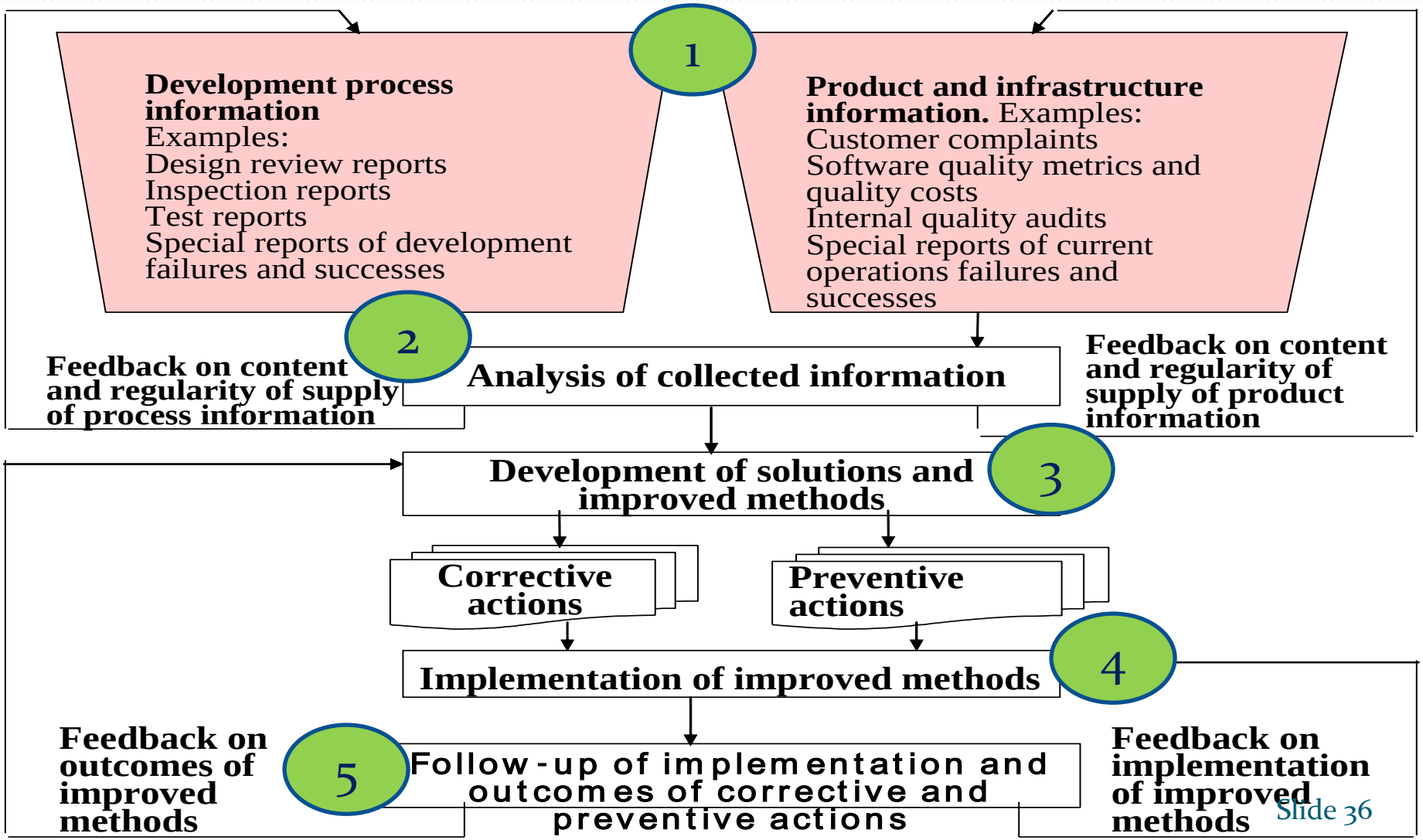
- Corrective and preventive actions (CAPA) — definitions
- The CAPA process
  - Information collection
  - Analysis of collected information
  - Development of solutions and their implementation
  - Follow-up of activities
  - Organizing for CAPA

CAPA - Definition

- Corrective Action: the action taken to eliminate the causes of an **existing** non-conformity, defect or other undesirable situation in order to prevent **recurrence**

- Preventive Action: action taken to eliminate the cause of a **potential** nonconformity, defect, or other undesirable situation in order to prevent **occurrence**

The CAPA process



The CAPA process

1. Sources of CAPA information

Internal information sources

Software development process

- Software risk management reports
- Design review reports
- Inspection reports
- Walkthrough reports
- Experts' opinion reports
- Test reviews
- Special reports on development failures and successes
- Proposal suggested by staff members.

Software maintenance

- Customer applications statistics
- Software change requests initiated by customer applications
- Software change requests initiated by maintenance staff
- Special reports on maintenance failures and successes
- Proposals suggested by staff members.

SQA infrastructure class of sources

- Internal quality audit reports
- External quality audit reports
- Performance follow-up of trained and certified staff
- Proposals suggested by staff members.

Software quality management procedures class of sources

- Project progress reports
- Software quality metrics reports
- Software quality cost reports
- Proposals of staff members.

External information sources

- Customer complaints
- Customer service statistics
- Customer-suggested proposals.

The CAPA process
2. Analysis of collected information

- Analysis involves:
  - screening the information and identifying potential improvements
  - analysis of potential improvements, to determine:
    - expected types and levels of damage
    - causes of faults
    - estimate total damage expected and determine the priority of each fault case
  - generating feedback on the content and regularity of information received from the designated information sources

The CAPA process
3. Development of solutions

- Several directions for solutions are commonly taken:
  - **updating relevant procedures**
  - changes in practices, including **updating of relevant work instructions**
  - **shifting to a development tool** that is more effective and less prone to the detected faults
  - **improvement of reporting methods**, including changes in report content, frequency of reporting and reporting tasks
  - initiatives for **training, retraining or updating staff**

The CAPA process
4. Implementation of the solutions

- Relies on proper instructions and often training but most of all on the cooperation of the relevant units and individuals

The CAPA process
5. Follow-up of activities

- Follow-up of the **flow of development and maintenance CAPA records** from various sources of information
  - enables feedback that reveals cases of no reporting, low quality reporting
- Follow-up of **implementation**
  - indicate whether the designated actions have been performed in practice
- Follow-up of **outcomes**
  - assessment of how much CAPA actions have achieved the expected results

Contents

- Procedures and work instructions
- Quality support devices
- Staff SQA training and certification activities
- Corrective and preventive actions
- **Software configuration management**
- Documentation control

Configuration management (CM)

- How to answer?
  - who can provide me with an accurate copy of last year's version 4.1 of the TMY software system?
  - what changes have been introduced in the new version of the software?
  - etc..
- Software configuration management (SCM) is the SQA component assigned to manage changes and supply accurate answers to inquiries of the types mentioned above

Sample: TDSys_CM plan_v1 0.doc

Terminology

- Software Configuration Item (SCI or CI):
  - Anything associated with a software project (design, code, test data, document, etc.) that has been placed under configuration control
  - Common types of SCI
    - Design documents
      - SDP, SRD, PDD, CDD, STP, STPR, STR, etc.
    - Software code
      - source code, object code, prototype software
    - Data file
      - test cases and test scripts, parameters, codes, etc.
    - Software development tools (the versions applied in the development and maintenance stages)
      - compilers and debuggers, application generators, CASE tools

Terminology

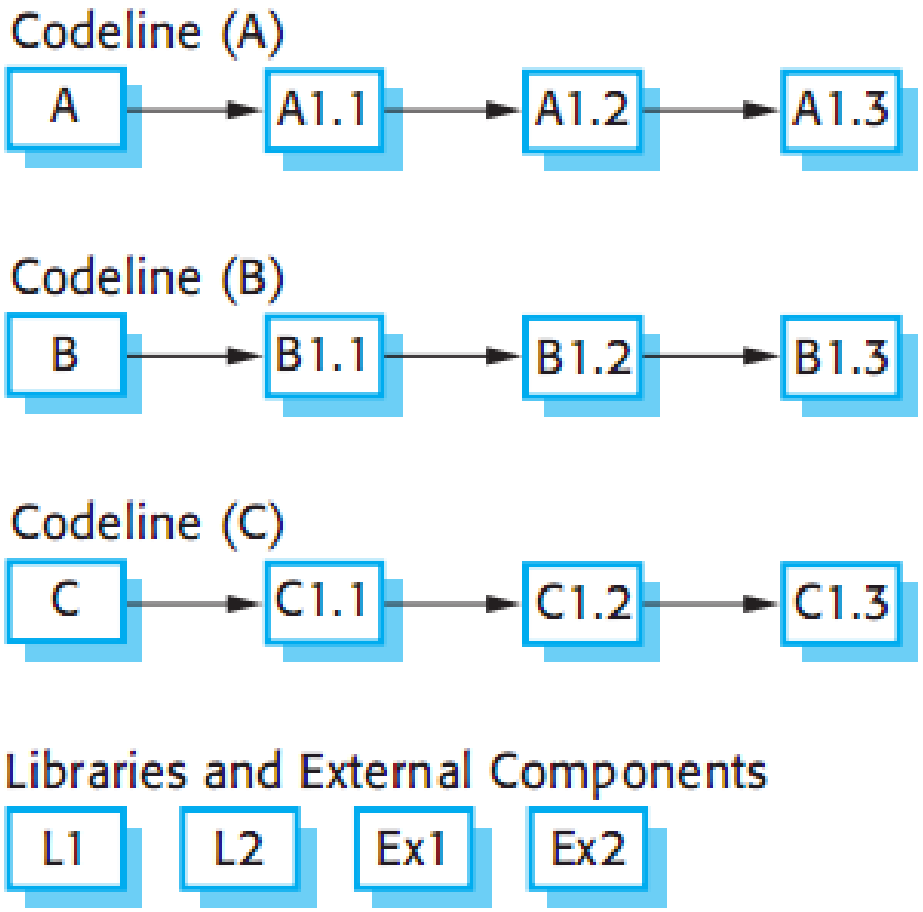
- Version
  - An instance of a CI that differs, in some way, from other instances of that item. Versions always have a unique identifier, which is often **composed of the configuration item name plus a version number**
- Release
  - A version of a system that has been released to customers (or other users in an organization) for use

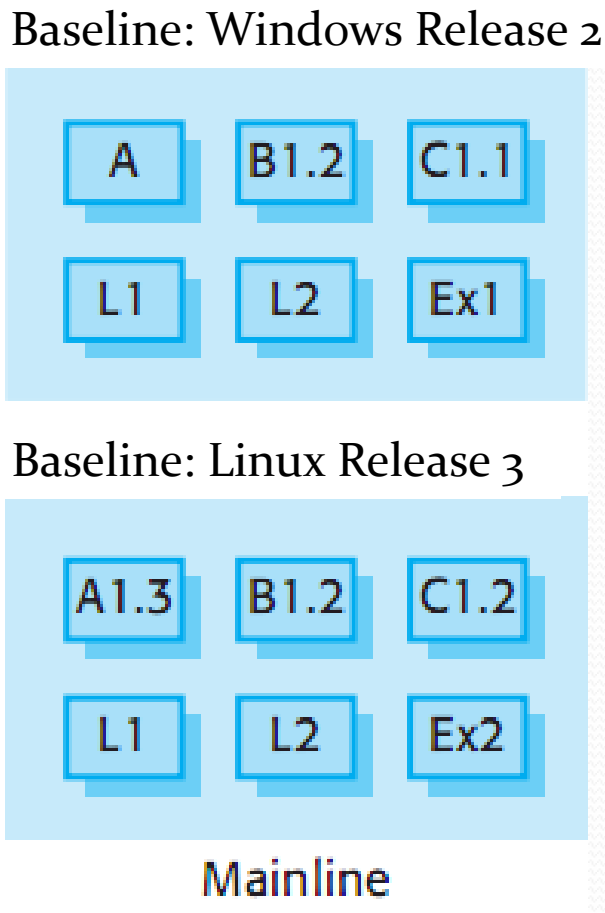
Terminology

- Codeline
  - A codeline is a **set of versions** of a software component and other configuration items on which that component depends
- Baseline
  - A baseline is a **collection of component versions** that make up a system
- Workspace
  - A **private work area** where software can be modified without affecting other developers who may be using or modifying that software

Terminology

Codelines and Baselines

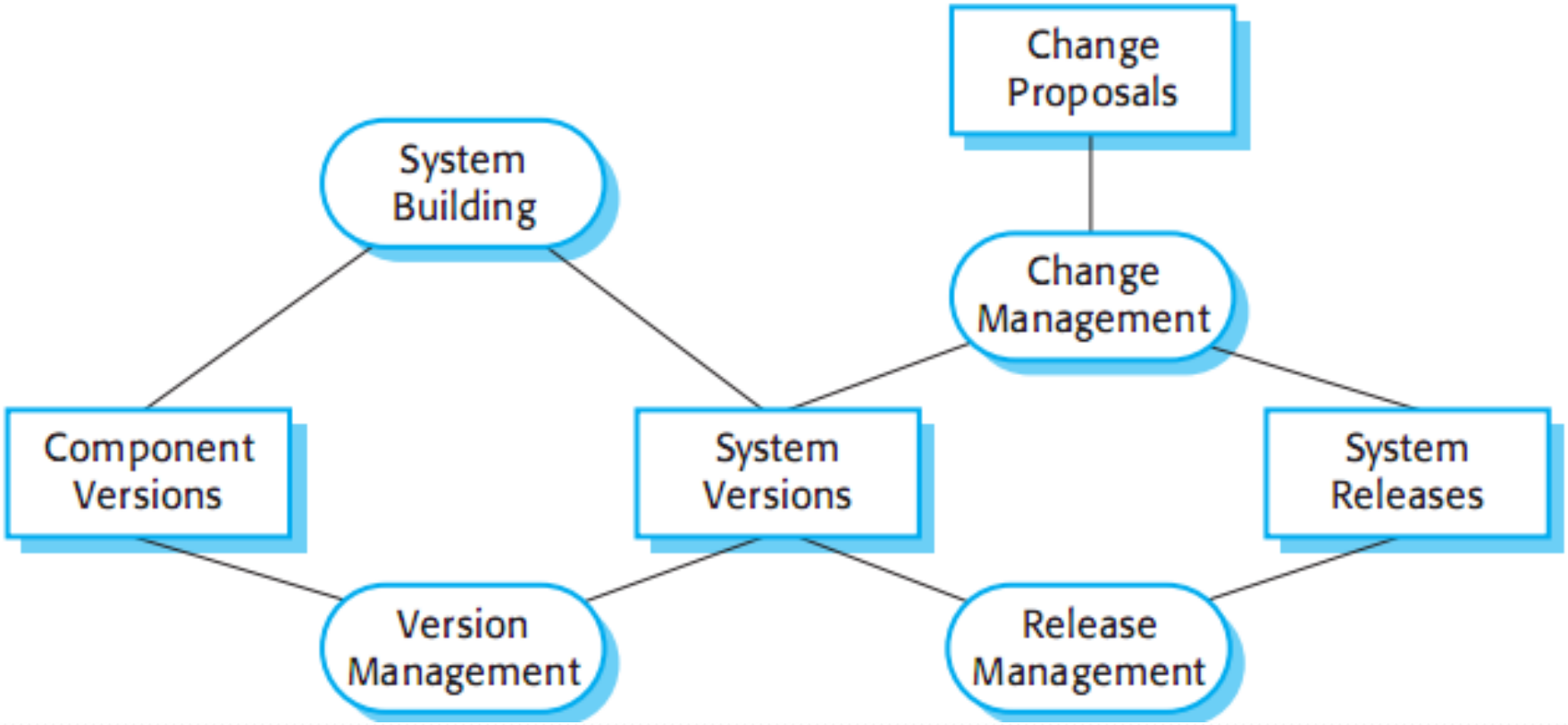




SCM activities

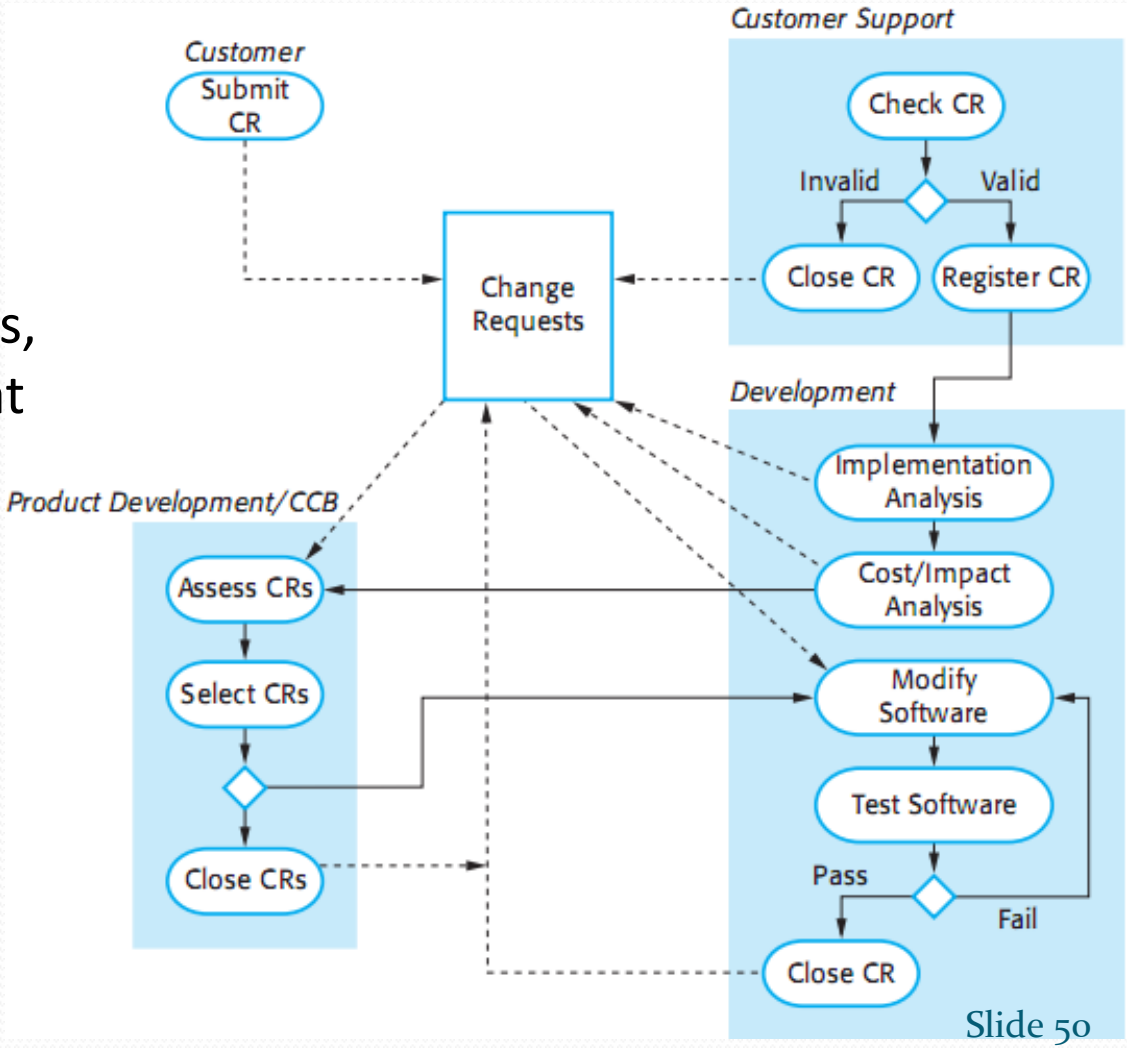
- Change management (Change control)
  - **Keeping track of requests for changes** to the software from customers and developers
- Version management (Version control)
  - **Keeping track of the multiple versions** of system components and ensuring that changes made to components by different developers do not interfere with each other
- System building
  - The **process of assembling** program components, data and libraries, then compiling these to create an executable system
- Release management
  - Preparing software for external release and **keeping track of the system versions** that have been released for customer use

SCM activities



Change management
The process

The change management process is concerned with analyzing the **costs** and **benefits** of proposed changes, **approving** those changes that are worthwhile and **tracking** which components in the system have been changed



A partially completed change request form (a)

Change Request Form

Project: SICSA/AppProcessing **Number:** 23/02
Change requester: I. Sommerville **Date:** 20/01/09
Requested change: The status of applicants (rejected, accepted, etc.) should be shown visually in the displayed list of applicants.

Change analyzer: R. Looek **Analysis date:** 25/01/09
Components affected: ApplicantListDisplay, StatusUpdater

Associated components: StudentDatabase

A partially completed change request form (b)

Change Request Form

Change assessment: Relatively simple to implement by changing the display color according to status. A table must be added to relate status to colors. No changes to associated components are required.

Change priority: Medium
Change implementation:
Estimated effort: 2 hours
Date to SGA app. team: 28/01/09 **CCB decision date:** 30/01/09
Decision: Accept change. Change to be implemented in Release 1.2
Change implementor: **Date of change:**
Date submitted to QA: **QA decision:**
Date submitted to CM:
Comments:

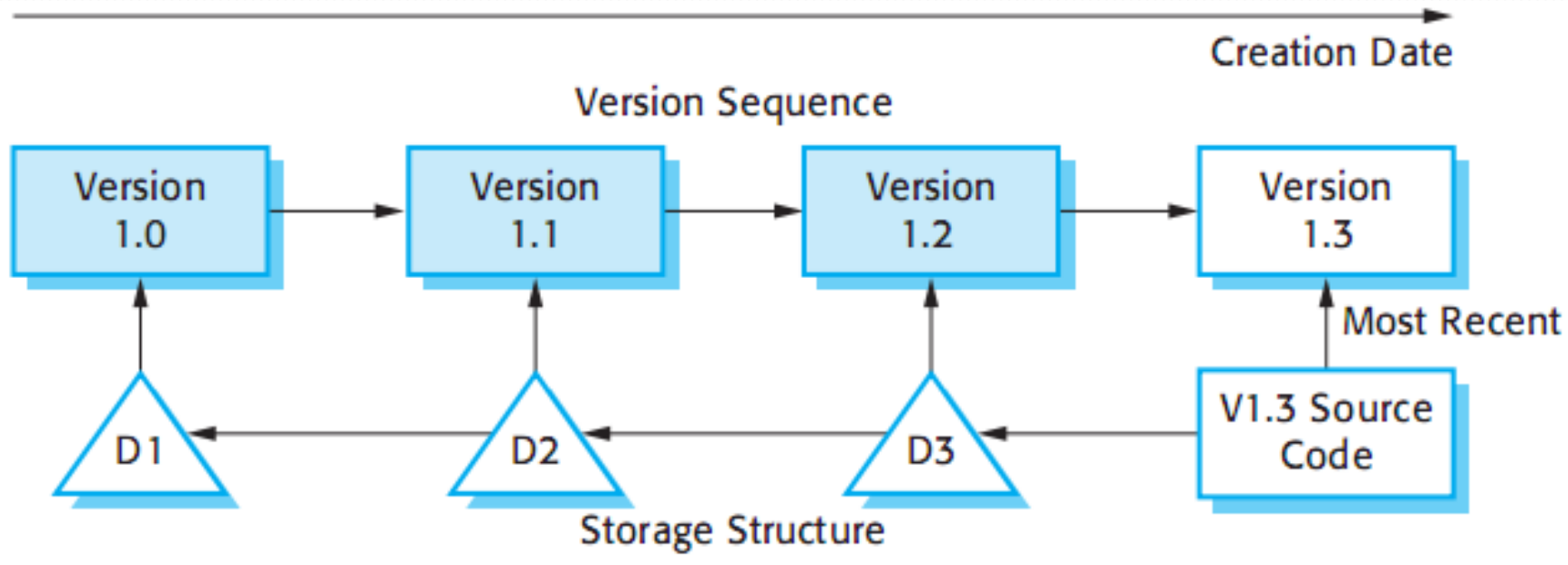
Version management

- Version management is the process of **keeping track of different versions** of software components or configuration items and the systems in which these components are used
- It also involves **ensuring that changes made by different developers to these versions do not interfere with each other**
- Therefore version management can be thought of as the process of managing **codelines** and **baselines**

Version management systems

- Version and release identification
  - Managed versions are **assigned identifiers** when they are submitted to the system
- Storage management
  - Instead of keeping a complete copy of each version, the system **stores a list of differences** (deltas) between one version and another
- Change history recording
  - All of the changes made to the code of a system or component are recorded and listed

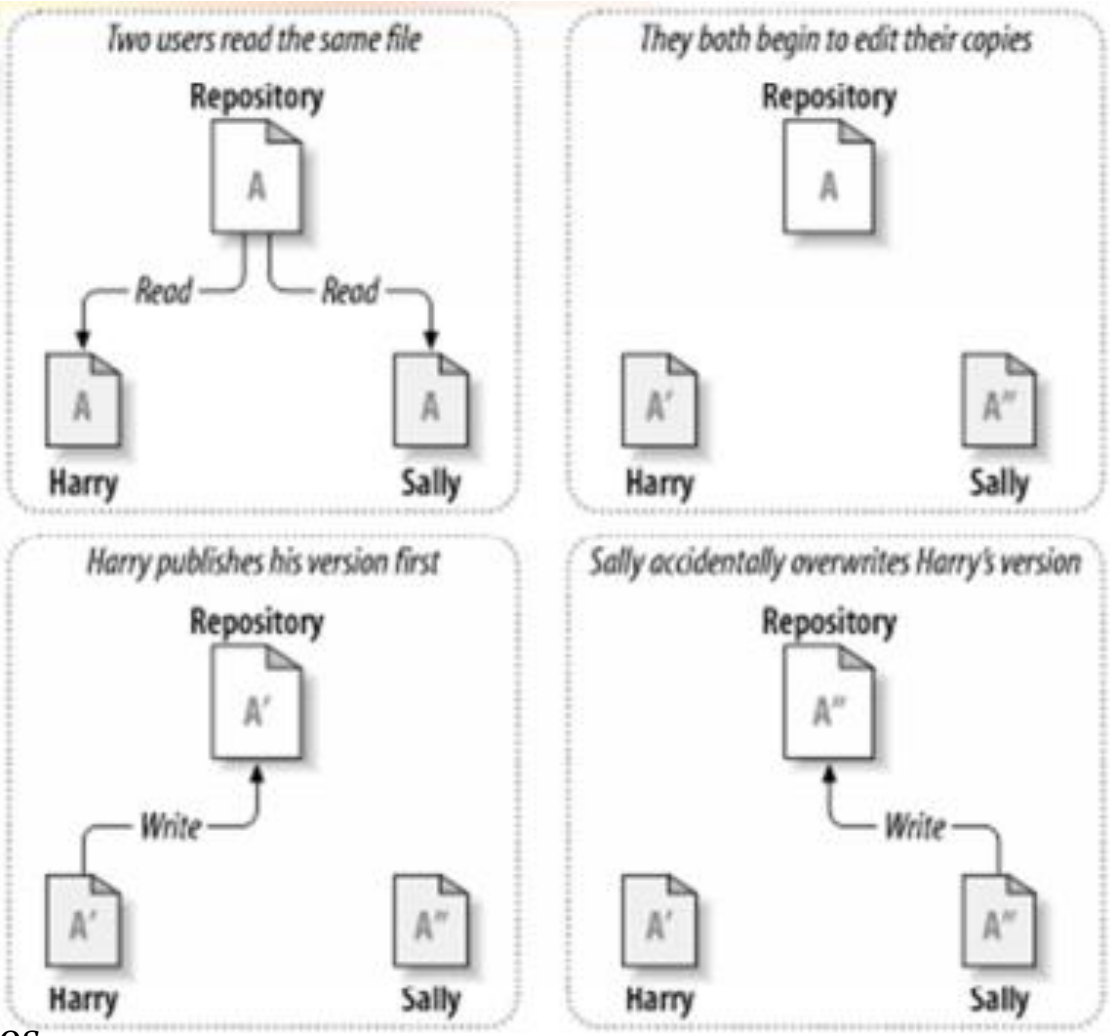
Version management systems
Storage management using deltas



Version management systems

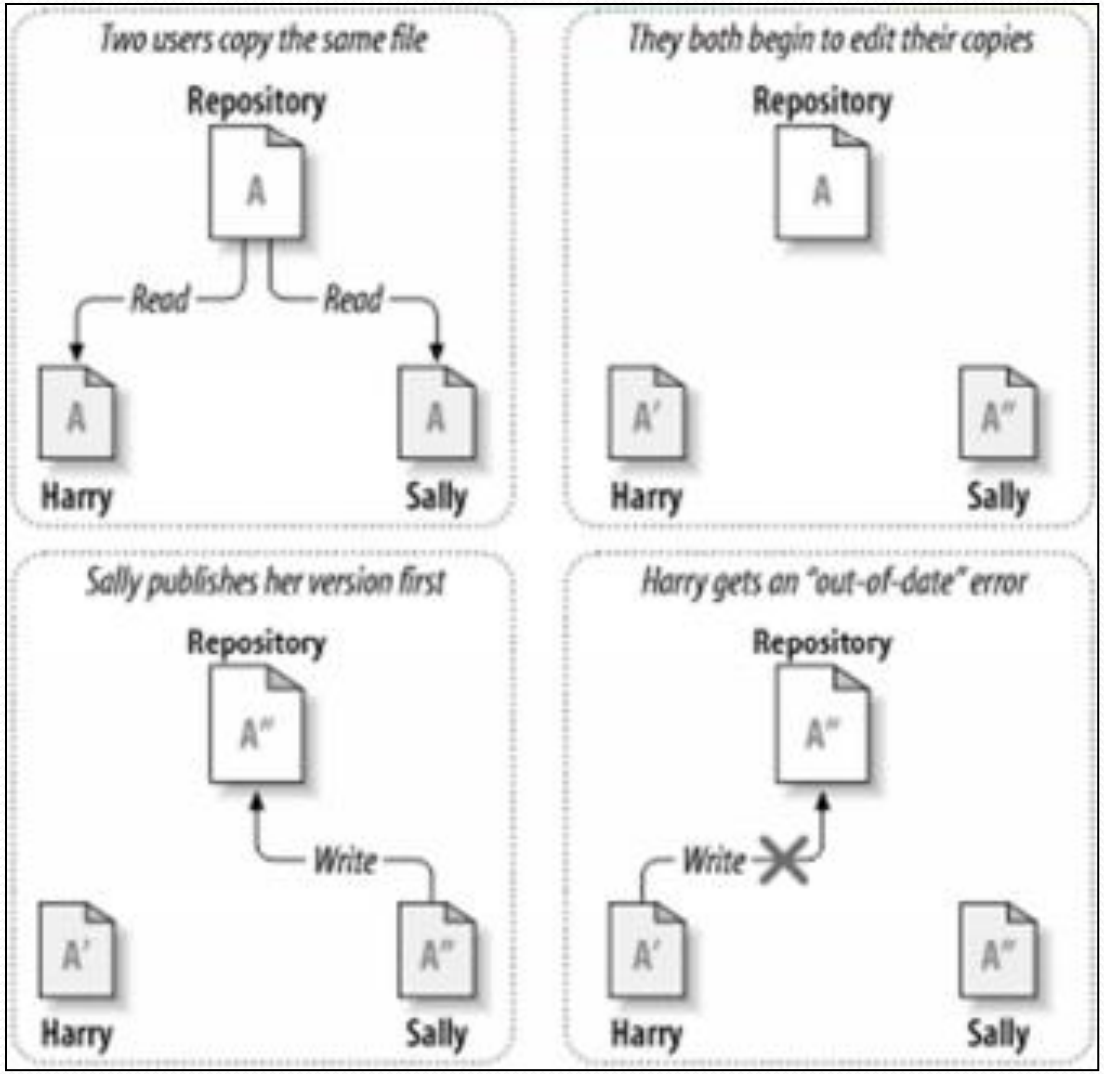
- Independent development
  - The version management system keeps track of components that have been checked out for editing and ensures that **changes made** to a component **by different developers** do **not interfere**
- Project support
  - A version management system may support the development of several projects, which **share components**

Version management systems
The problem of file sharing



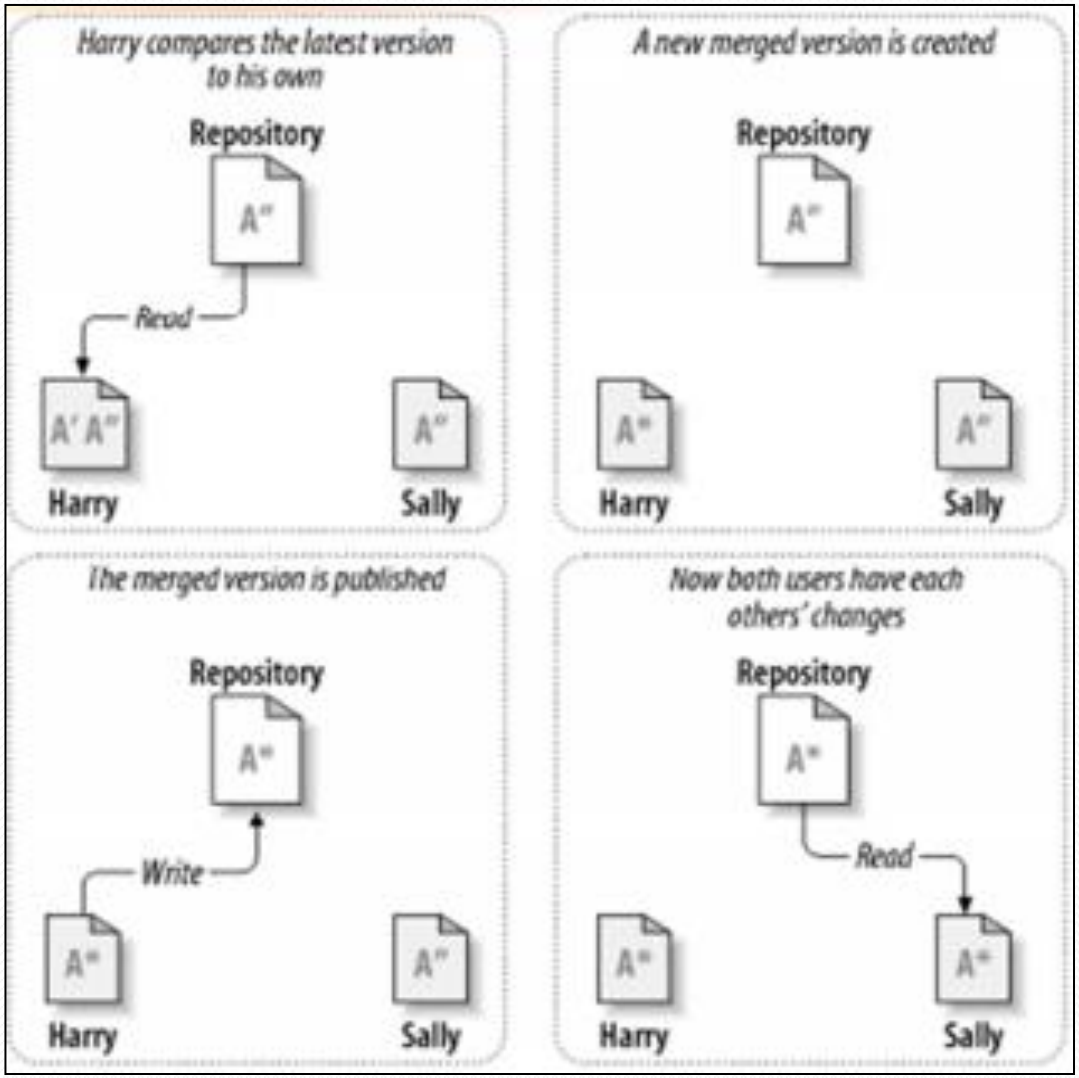
From FSOFT slides

Version management systems
The solution of file sharing



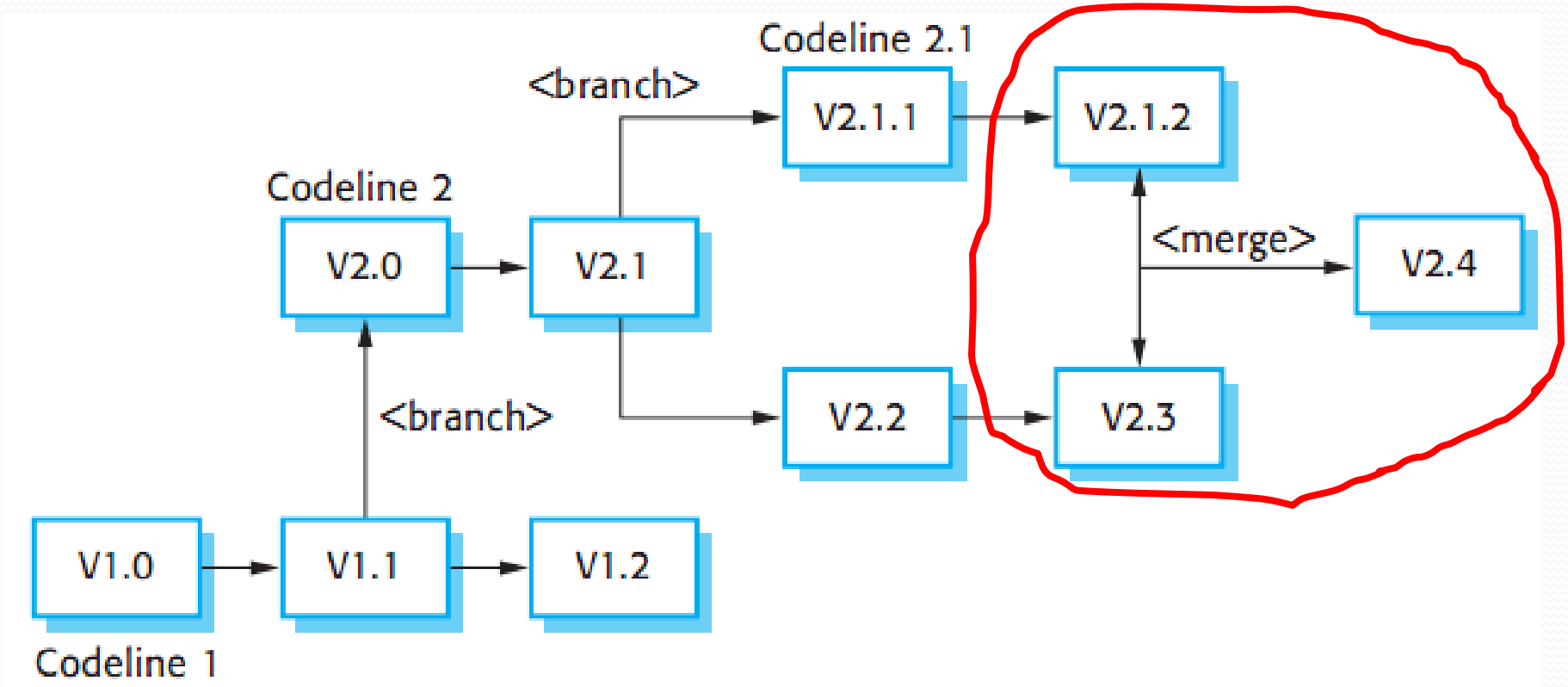
From FSOFT slides

Version management systems
The solution of file sharing



From FSOFT slides

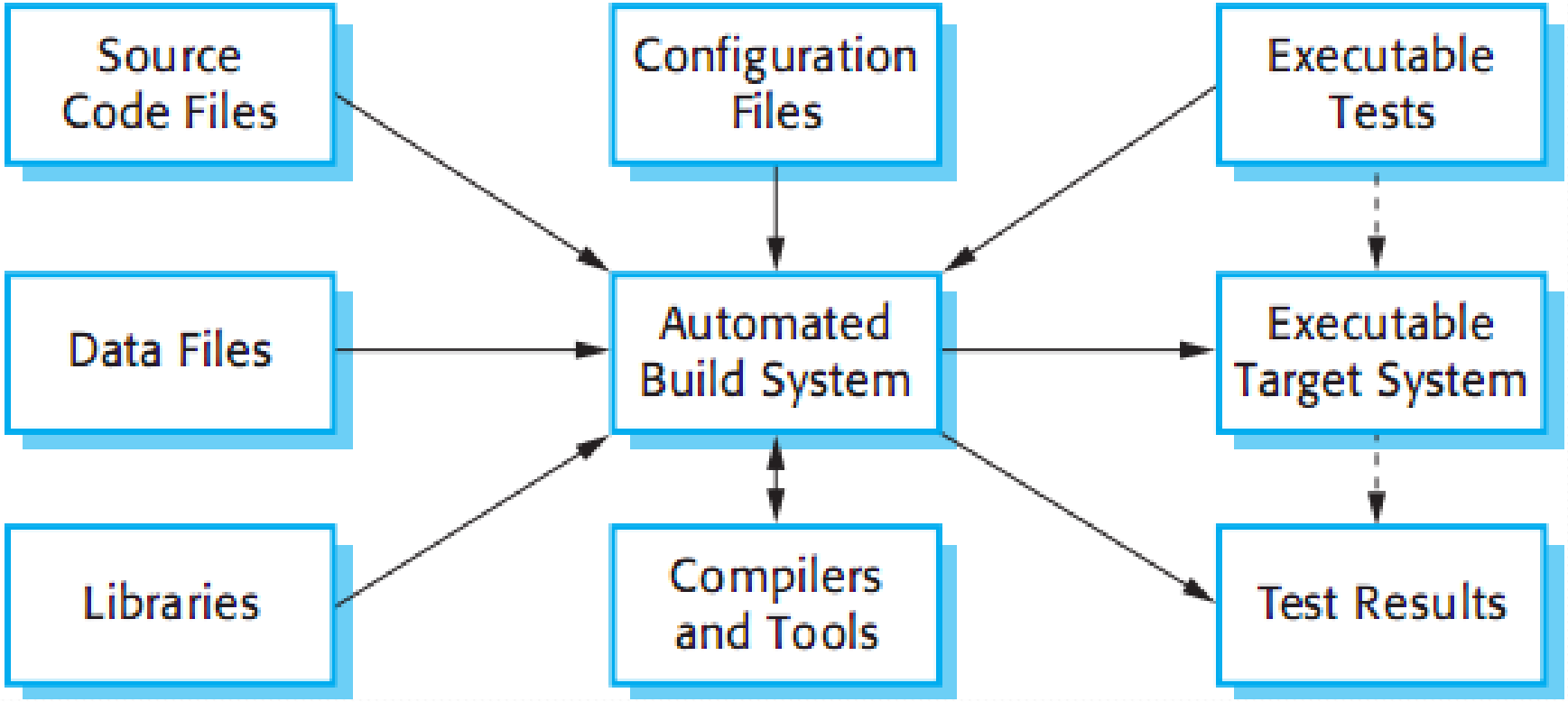
Version management systems
Branching and merging



System building

- System building is the process **of creating a complete, executable system** by compiling and linking the system components, external libraries, configuration files, etc.
- System building tools and version management tools must communicate as the build process involves checking out component versions from the repository managed by the version management system

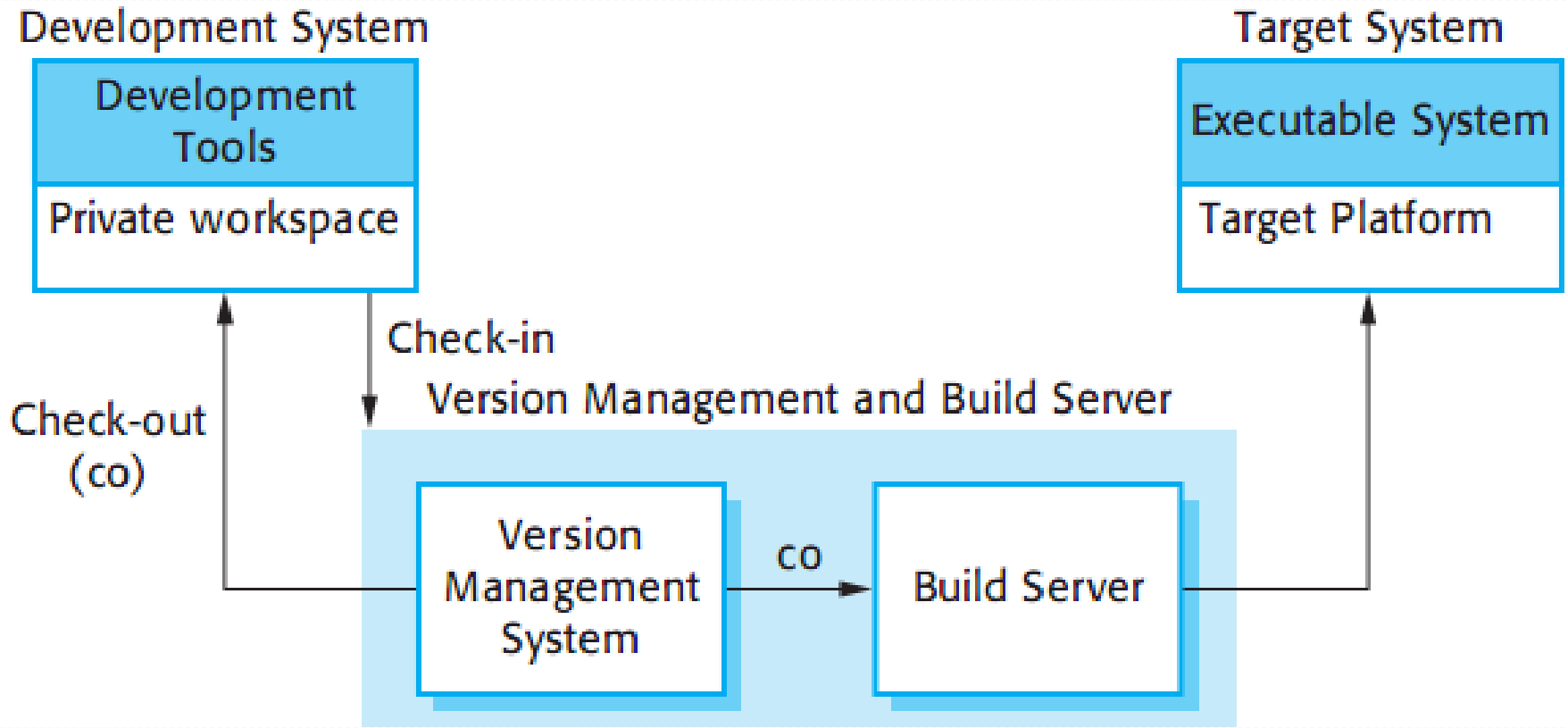
System building



System building
Build platforms

- **The development system**, which includes development tools such as compilers, source code editors, etc.
  - Developers check out code from the version management system into a private workspace before making changes to the system
- **The build server**, which is used to build definitive, executable versions of the system
  - Developers check-in code to the version management system before it is built. The system build may rely on external libraries that are not included in the version management system
- **The target environment**, which is the platform on which the system executes

System building
Development, build, and target platforms



Release management
Release tracking

- In the event of a problem, it may be necessary to **reproduce exactly the software** that has been delivered to a particular customer
- When a system release is produced, **it must be documented** to ensure that it can be re-created exactly in the future

Release management
Release reproduction

- To document a release, you have to **record the specific versions of the source code** components that were used to create the executable code
- You should also record the versions of the **operating system, libraries, compilers and other tools** used to build the software
- You must **keep copies** of the source code files, corresponding executables and all data and configuration files

Release management
Release components

- As well as the the executable code of the system, a release may also include:
  - configuration files defining how the release should be configured for particular installations;
  - data files, such as files of error messages, that are needed for successful system operation;
  - an installation program that is used to help install the system on target hardware;
  - electronic and paper documentation describing the system;
  - packaging and associated publicity that have been designed for that release

Contents

- Procedures and work instructions
- Quality support devices
- Staff SQA training and certification activities
- Corrective and preventive actions
- Configuration management
- **Documentation control**

Documentation control

- Controlled document
  - a document that is currently vital or may become **vital for the development and maintenance** of software systems as well as **for the management of current and future relationships with the customer**
- **Documentation procedures** control its preparation, storage, retrieval and disposal

Typical controlled documents

- Pre-project documents
  - contract review report, negotiation meeting minutes, development contract, subcontracting contract, software development plan, etc.
- Project life cycle documents
  - SRD, PDD, CDD, database description, DR report, STP, etc.
- SQA infrastructure documents
  - SQA procedures, template library, SQA form library, etc.
- Software quality management documents
  - progress report, software metrics reports, etc.
- SQA audit documents
  - management review report, internal quality audit report, etc.
- Customer documents
  - software project tender documents, customer's software change requests, etc.

Documentation control - objectives

- To **assure the quality** of the document
- To **assure its technical completeness and compliance** with document structure procedures and instructions (use of template, proper signing, etc)
- To **assure the future availability** of documents that may be required for software system maintenance, further development, or responses to the customer's (tentative) future complaints
- To **support investigation of software failure causes** and to **assign responsibility** as part of corrective and other actions

Typical components of documentation control procedures

- Definition of the **list of the document types** and updates to be controlled (some classified as quality records)
- Document **preparation requirements**
- Document **approval requirements**
- Document **storage** and **retrieval requirements**, including controlled storage of document versions, revisions and disposal, document security

The controlled documents list

- Authority for controlled document and quality record list
  - Deciding which document type is to be categorized as a **controlled document** and which controlled document types are to be classified as **quality records**
  - Deciding whether the **level of control** is adequate for each document type categorized as a controlled document
  - **Following up** of compliance with the controlled document types lists. This can be incorporated in the internal quality plan
  - **Analyzing** follow-up findings and initiating the required updates, changes, removals and additions to the controlled documents types list

Controlled document preparation

- **Creation** of new document or **revision** of an existing document focus on completeness, improved readability and availability
- This relies in the document:
  - structure – may be free or defined by a template
  - identification method – usually entails notation of (a) the software system or product name or number, (b) the document (type) code and (c) the version and revision number
  - standard orientation and reference information – support future access

Issues of controlled document approval

- Position of the person(s) who can approve a document or document type
  - can be granted by a person, several persons, or committees
  - have sufficient experience and technical expertise

- The approval process
  - aim at detecting and preventing professional inadequacies together with deviations from the document template

Issues of controlled document storage and retrieval

- Document storage
  - number of copies, unit responsible, storage medium
- Circulation and retrieval of documents
  - instruction for circulating a new document, on time, recipients; efficient and accurate retrieval of copies, in full compliance with security restrictions
- Document security, including document disposal requirement
  - provide restricted access; prevent unauthorized changes to stored documents; provide back-up; determine storage period