SOFTWARE ENGINEERING
LECTURE 9

CARSTEN RUSENG JAKOBSEN

AALBORG UNIVERSITY
DENMARK

CONFIGURATION MANAGEMENT

Configuration Management (CM) particulars

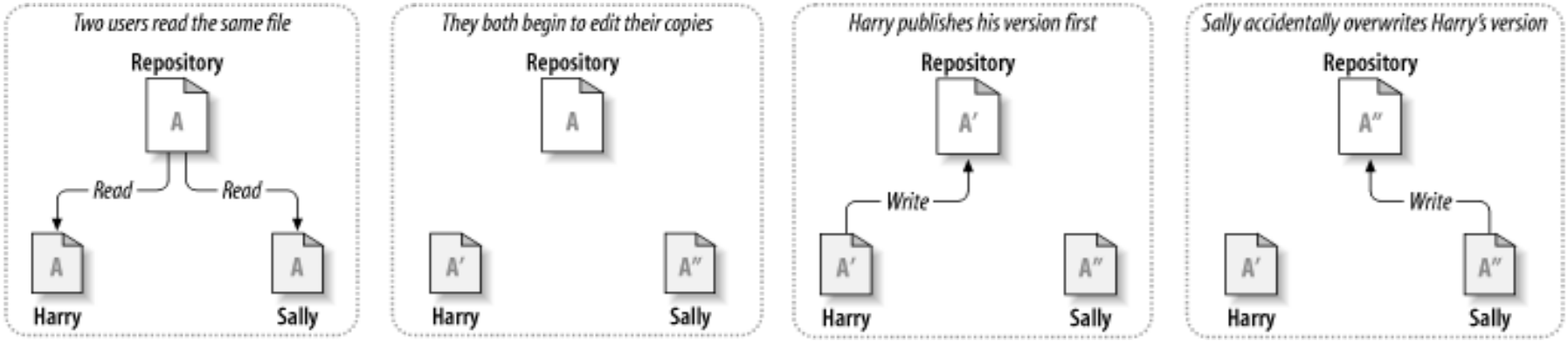
… CM is concerned with the policies, processes and tools for managing changing software systems.

Involves four closely related activities:

1. Version management
2. System building
3. Change management
4. Release management

- Plan-driven vs agile CM

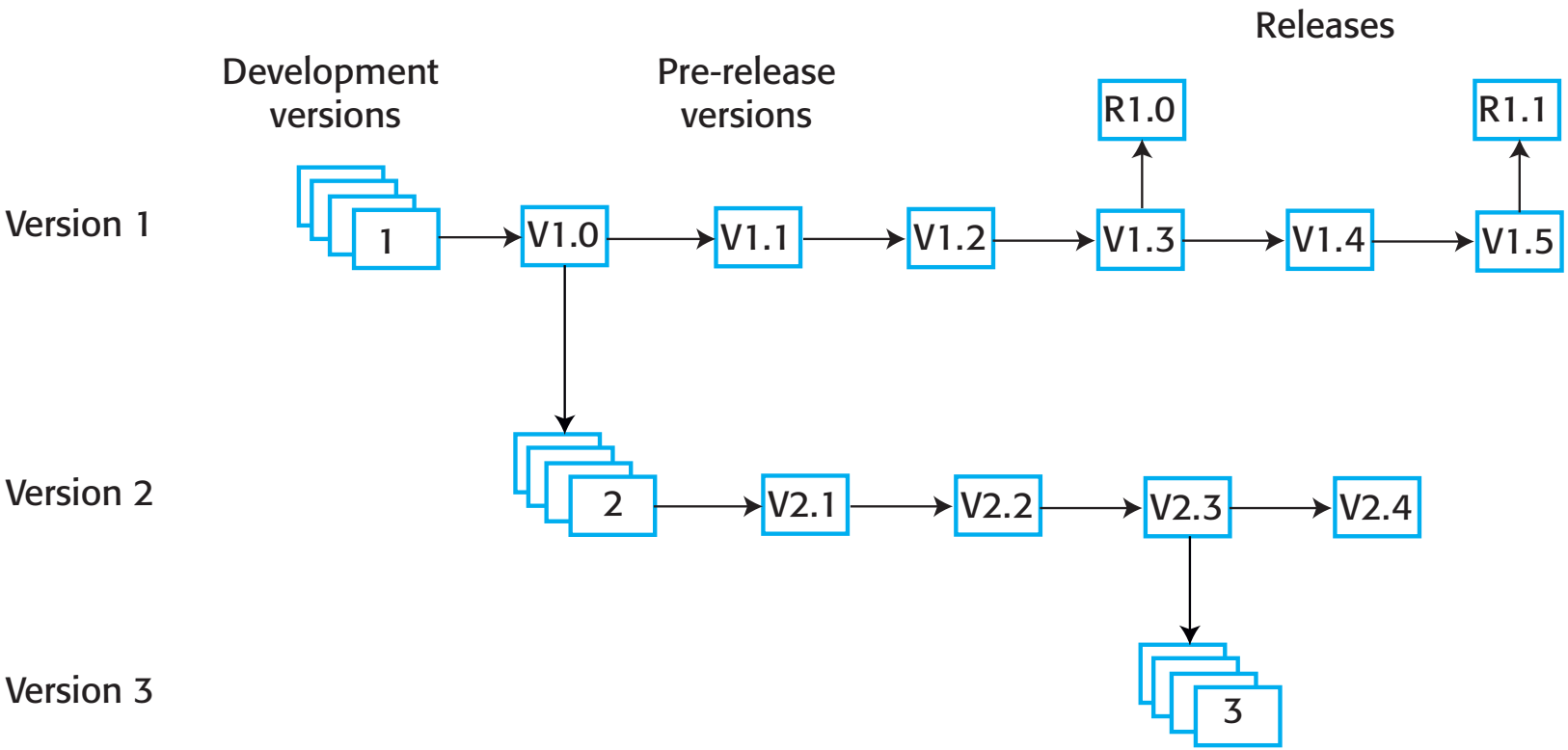
Managing collaborative working: Version & revision control



Solutions:

- Pessimistic: file locking
- Optimistic: version merging

Multi-version system development



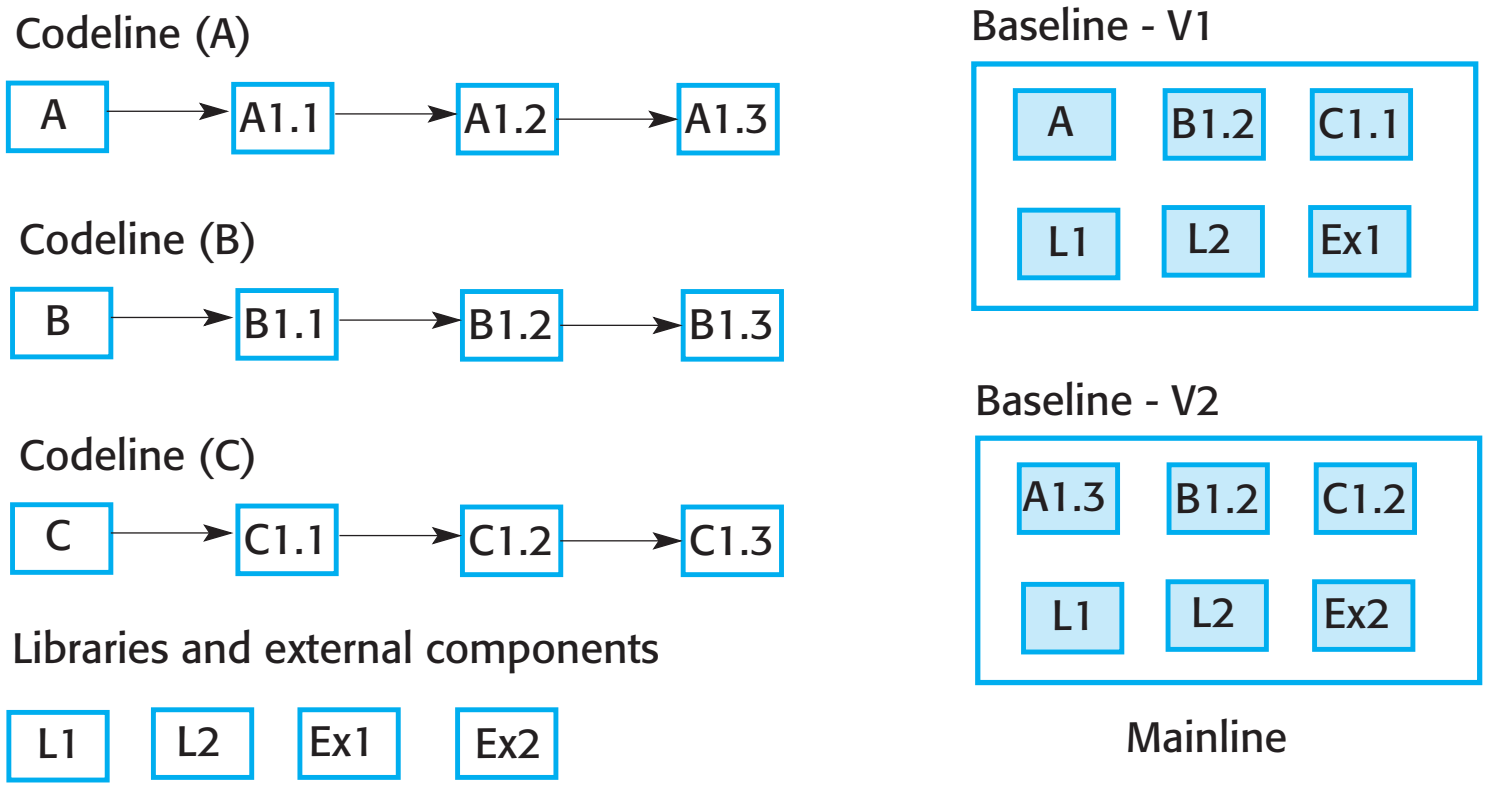
CM terminology

Term	Explanation
Baseline	A baseline is a collection of component versions that make up a system. Baselines are controlled, which means that the versions of the components making up the system cannot be changed. This means that it is always possible to recreate a baseline from its constituent components.
Branching	The creation of a new codeline from a version in an existing codeline. The new codeline and the existing codeline may then develop independently.
Codeline	A codeline is a set of versions of a software component and other configuration items on which that component depends.
Configuration (version) control	The process of ensuring that versions of systems and components are recorded and maintained so that changes are managed and all versions of components are identified and stored for the lifetime of the system.
Configuration item or software configuration item (SCI)	Anything associated with a software project (design, code, test data, document, etc.) that has been placed under configuration control. There are often different versions of a configuration item. Configuration items have a unique name.
Mainline	A sequence of baselines representing different versions of a system.

CM terminology

Term	Explanation
Merging	The creation of a new version of a software component by merging separate versions in different codelines. These codelines may have been created by a previous branch of one of the codelines involved.
Release	A version of a system that has been released to customers (or other users in an organization) for use.
Repository	A shared database of versions of software components and meta-information about changes to these components.
System building	The creation of an executable system version by compiling and linking the appropriate versions of the components and libraries making up the system.
Version	An instance of a configuration item that differs, in some way, from other instances of that item. Versions always have a unique identifier.
Workspace	A private work area where software can be modified without affecting other developers who may be using or modifying that software.

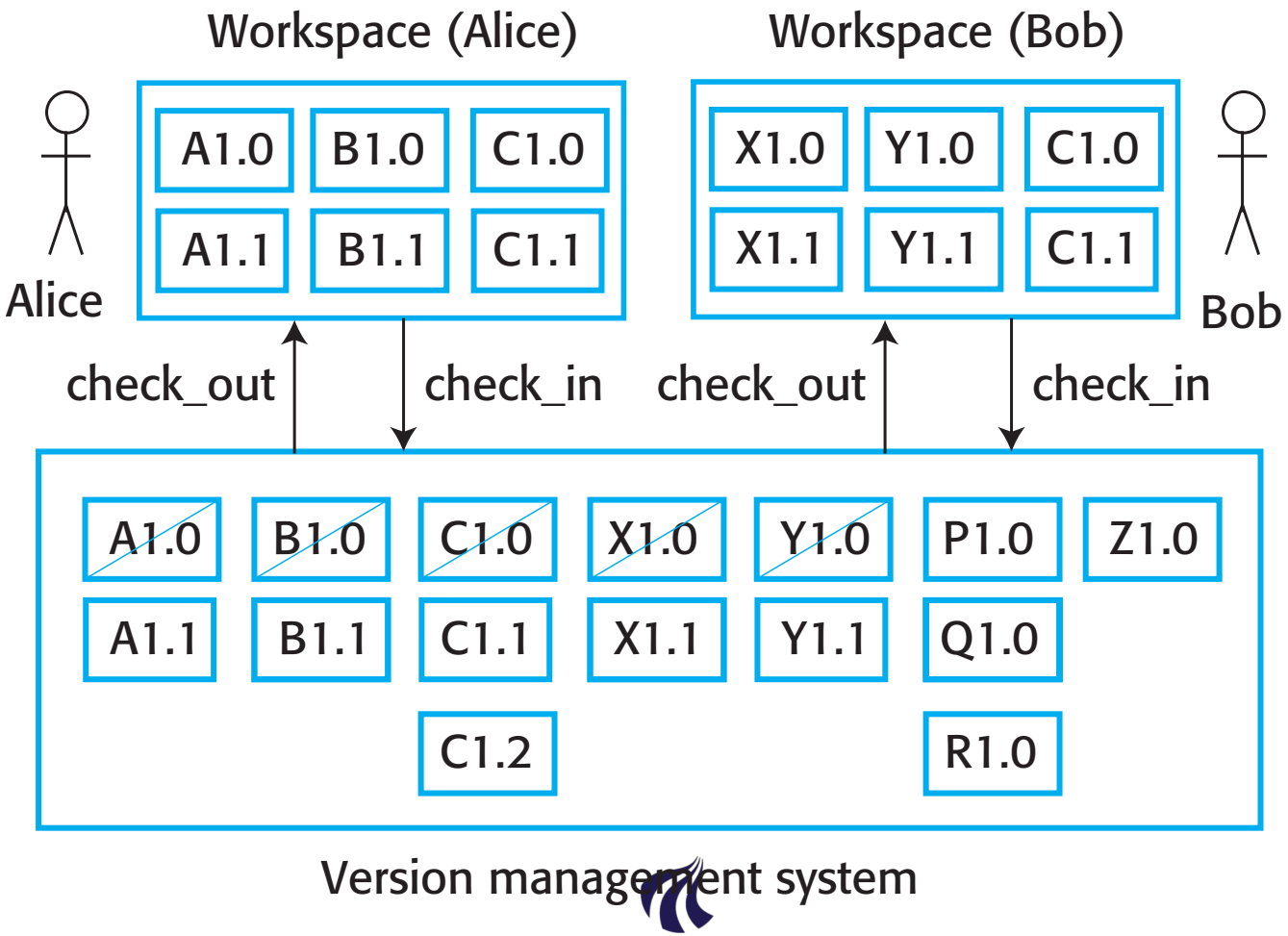
Codelines and baselines



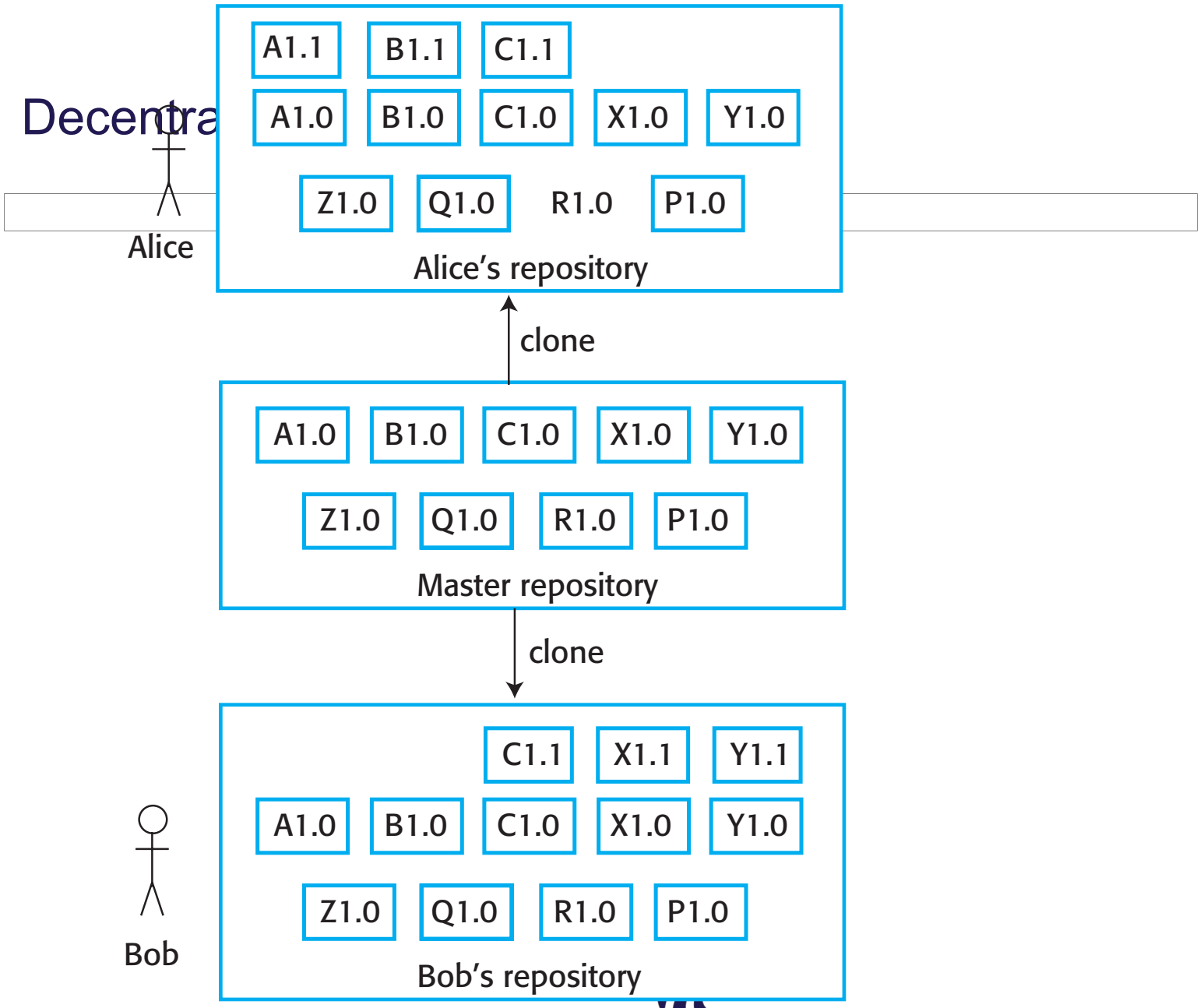
Version control systems

- Version control (VC) systems identify, store and control access to the different versions of components. There are two types of modern version control system
  - Centralized systems, where there is a single master repository that maintains all versions of the software components that are being developed. Subversion is a widely used example of a centralized VC system.
  - Distributed systems, where multiple versions of the component repository exist at the same time. Git is a widely-used example of a distributed VC system.

Centralised: Check-in/Check-out



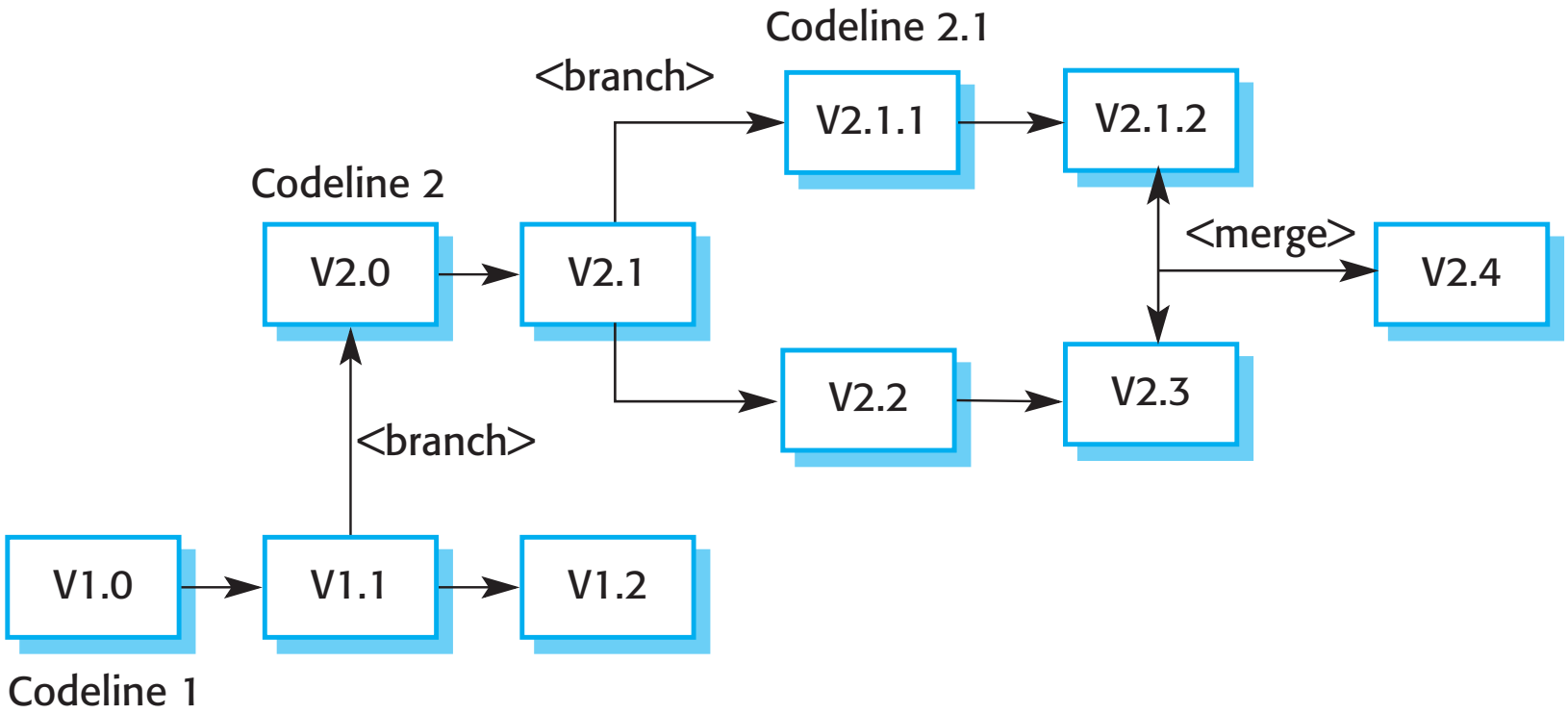
Decentra


A1.1
B1.1
C1.1
A1.0
B1.0
C1.0
X1.0
Y1.0
Z1.0
Q1.0
R1.0
P1.0
Alice
Alice's repository
clone
A1.0
B1.0
C1.0
X1.0
Y1.0
Z1.0
Q1.0
R1.0
P1.0
Master repository
clone
C1.1
X1.1
Y1.1
A1.0
B1.0
C1.0
X1.0
Y1.0
Z1.0
Q1.0
R1.0
P1.0
Bob
Bob's repository

Benefits of distributed version control

- It provides a backup mechanism for the repository.
  - If the repository is corrupted, work can continue and the project repository can be restored from local copies.
- It allows for off-line working so that developers can commit changes if they do not have a network connection.
- Project support is the default way of working.
  - Developers can compile and test the entire system on their local machines and test the changes that they have made.
- Essential in open source development! Why?

Branching and merging



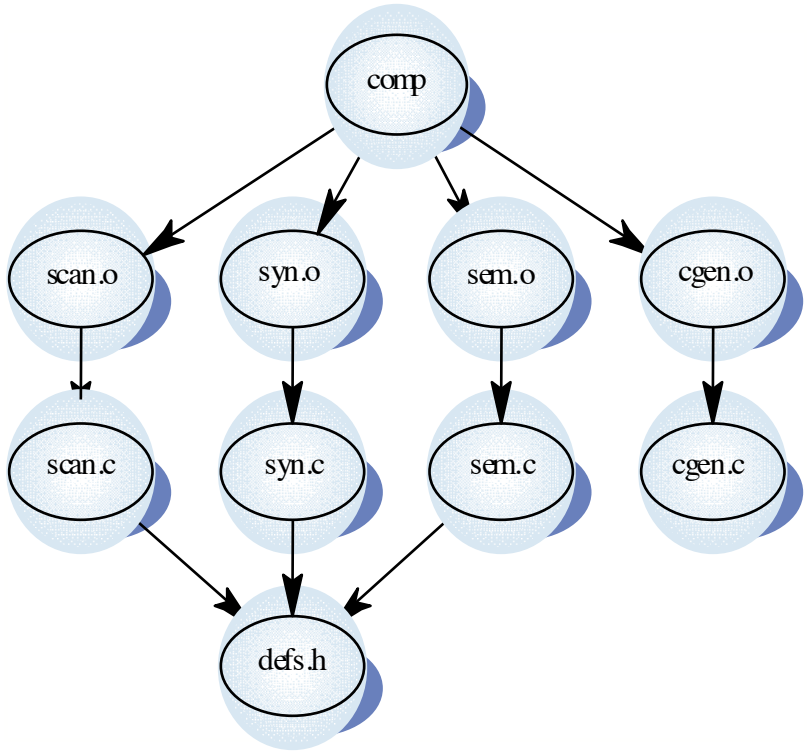
System building tools

Building a large system is computationally expensive and may take several hours

Hundreds of files may be involved

System building tools may provide:

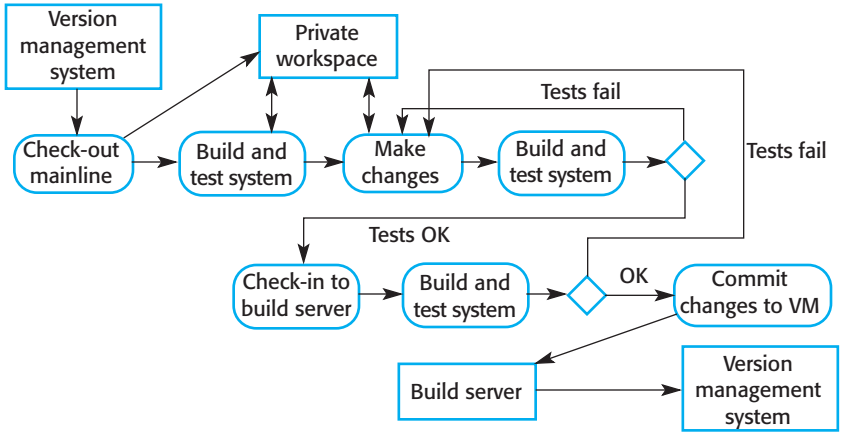
- a dependency specification language and interpreter
- tool selection and instantiation support
- distributed compilation



Examples:

Scons, Jenkins CI

Continuous integration



Daily building

- The development organization sets a delivery time (say 2 p.m.) for system components.
  - If developers have new versions of the components that they are writing, they must deliver them by that time.
  - A new version of the system is built from these components by compiling and linking them to form a complete system.
  - This system is then delivered to the testing team, which carries out a set of predefined system tests
  - Faults that are discovered during system testing are documented and returned to the system developers. They repair these faults in a subsequent version of the component.

Plan-driven vs agile configuration management

AALBORG UNIVERSITY
DENMARK

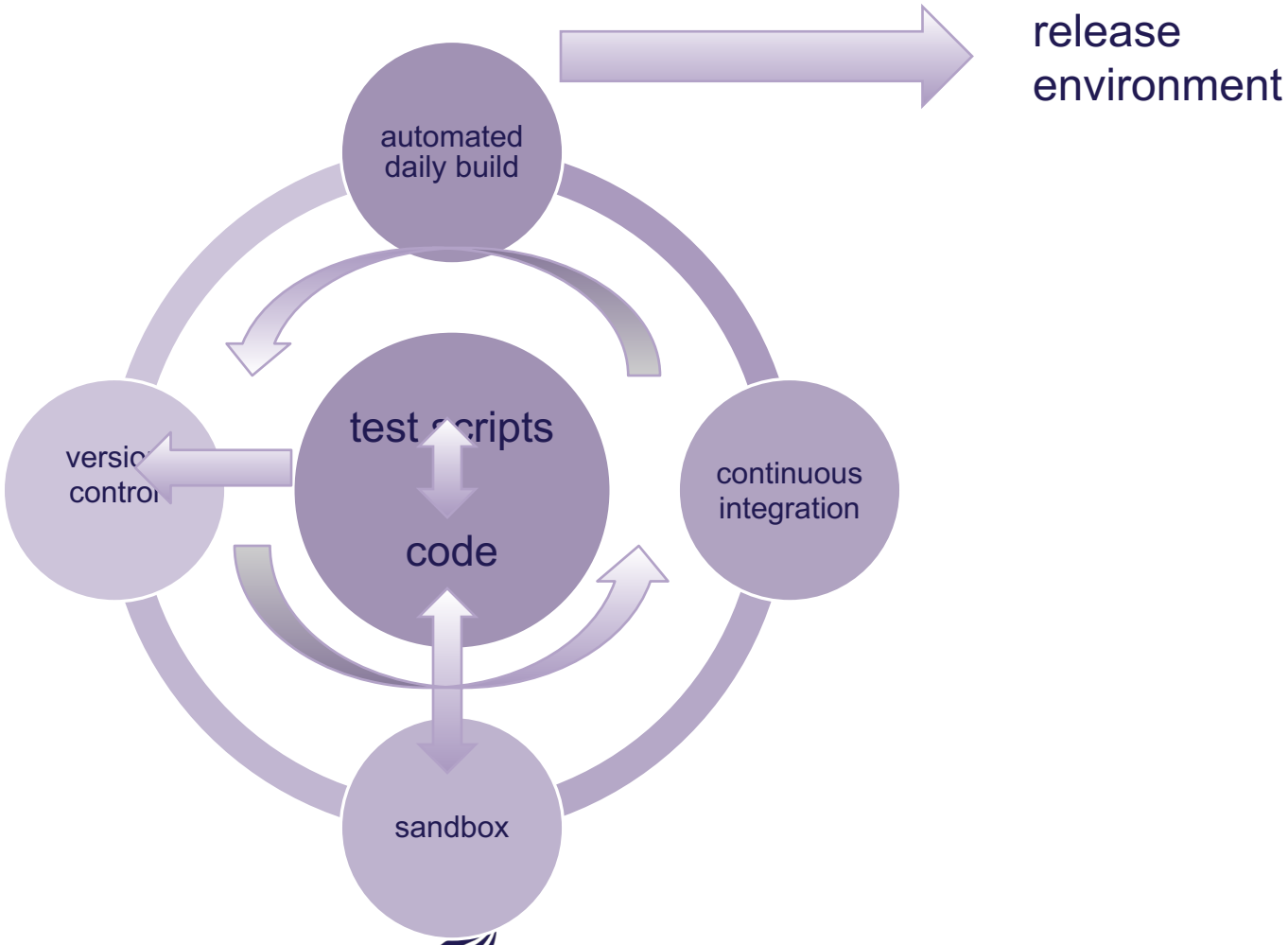
Plan-driven configuration management

- Configuration management is the organizational and tool supported management of change to software products
- Formal document naming schemes and management in a database
- The configuration database should record information about changes and change requests
- A consistent scheme of version identification should be established using version numbers, attributes or change sets
- System releases include executable code, data, configuration files and documentation
- System building involves assembling components into a system
- Tools are available to support all CM activities
- Tools may be stand-alone or may be integrated systems which integrate support for version management, system building and change management

Agile development and CM

- Agile development, where components and systems are changed several times per day, is impossible without using CM tools.
- The definitive versions of components are held in a shared project repository and developers copy these into their own workspace.
- They make changes to the code then use system building tools to create a new system on their own computer for testing. Once they are happy with the changes made, they return the modified components to the project repository.

Agile Configuration Management



Agile Configuration Management

Schuh, P. (2007) Agile Configuration management for large organizations. In: *The Rational Edge, IBM*.

- Build automation
- Automated migration and deployment
- Test automation
- Continuous integration
- Corporate CM organization

Plan-driven and agile configuration management

	Plan-driven	Agile
Focus	documents and code	code
Activities	CM practice, change management, version control, automated build	version control, automated build
Responsible	CM team, CM board	programmers
Process	formal, managed	informal and integrated with practice environment
Outcome	CM audit (documented product control)	next release
Importance	indispensible in medium and large projects	indispensible