




SYSTEMS PLANNING


Software Configuration Management

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu


1



Class content

What to expect from this class?

- An **introduction to Software Configuration Management** and its processes
- The **purpose of Change Management**
- The **purpose of Version Management**
- The **purpose of System Building**
- The **purpose of Release Management**



Source: <https://pixabay.com/en/checklist-check-list-marker-2077020/>
by TerriKoskunen / CC0 Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

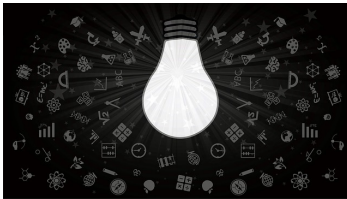
2

Class learning goals



What should be your learning outcome?

- Know the **stages of Software Configuration Management**
- Understand the **goals of (Software) Change Management**
- Understand the **goals of Version Management**
- Understand the **process of System Building**
- Understand the **challenges of Release Management**



Source: <https://pixabay.com/en/learning-hint-school-subject-3245792/>
by harishs | CC0 Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

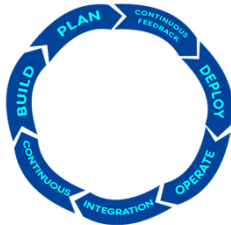
3

Software Configuration Management



Software Change

- Software systems always **change**
- **Bugs** are discovered, **system requirements** change, new **versions** of hard- and software become available, and **competitors** introduce new features
- Configuration Management (CM) deals with **policies, processes and tools** for managing changing software systems
- Often there may be **several versions** of a software under development and in use
- CM helps to **keep track** of these versions



Source: <https://pixabay.com/en/isolated-objects-business-3188337/>
by dlowu | CC0 Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

4

Software Configuration Management




Configuration Management

- Configuration Management is useful for projects as individuals may **forget what changes have been made**
- It is essential for team projects where increasingly members (i.e. programmers) are distributed in **different locations** around the world

Configuration management includes:

- **Change management:** Keep track of changes
- **Version management:** Keep track of versions
- **System building:** assembling components, data and libraries
- **Release management:** Preparing software for external release



Source: <https://pixabay.com/en/gear-engine-config-icon-2933357/>
by raphaeltiva | CC0 Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu

5

Software Configuration Management




Change Management

- Organizational **needs and requirements** change
- Systems have to **adapt to changes in their environment**
- Tool-supported change management processes ensure **that system changes are controlled**

Change management ensures that

- **System evolution** is a **managed process**
- **Priority** is given to the **most urgent and cost-effective changes**
- all changes are **appropriately tracked**



Source: <https://pixabay.com/en/change-new-beginning-renewal-671374/>
by geralt | CC0 Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu


6

Software Configuration Management



Change Management Process


- The change management process is initiated when the 'customer' **submits a change request**
- This can be a **bug report** or a **request for change**
- Usually this happens through a **Change Request Form (CRF)**
- The CRF is a shared document and **constantly updated**
- It therefore represents a **snapshot of the current state** of the change request
- It records recommendations regarding the **change, estimated costs, and various dates** (e.g. request, approved, implemented, validated)
- It may also outline **how** a change is to be **implemented**



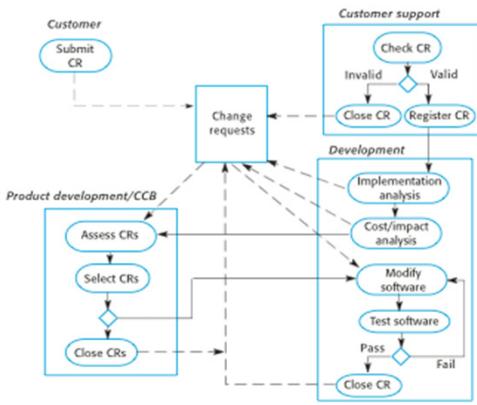
Source: <https://pixabay.com/en/bulb-nature-ecologycurrent-2368396/>
by thommas68 | CC0 Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
7

Software Configuration Management




Change Management Process



Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
8

Software Configuration Management



Change Request Form

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. Loeek **Analysis date:** 25/01/09
Components affected: ApplicantListDisplay, StatusUpdater
Associated components: StudentDatabase

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:

Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.


DIE UNTERNEHMERISCHE HOCHSCHULE®
 MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
 Universitätsstraße 15

www.mci.edu
 office@mci.edu


9

Software Configuration Management



Change Management Process

- Submitted change requests need to be **validated**
- **Not all changes require actions** (e.g. a bug may already be solved)
- Sometimes they are based on **misunderstandings** of what the system should actually do
- Sometimes people request **features that already exist**
- Valid requests are then **assessed** and their **cost estimated**
- The **impact** of change **on the rest of the system** needs to be checked
- Some changes **affect multiple components** which increases the overall cost



Source: <https://pixabay.com/en/cost-board-finance-money-business-1174926/>
by geralt | CC0 Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
 MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
 Universitätsstraße 15

www.mci.edu
 office@mci.edu


10

Software Configuration Management



Change Management Process

- A **separate group** should then **decide** if it is **cost-effective** from a business perspective to make the change to the software
- This **group approves all change requests** unless they are minor (e.g. errors on a screen display or webpage)
- The group **considers the impact of the change from a strategic** rather than a technical point of view
- **Accepted changes** are passed back to the **development team**, **rejected changes are closed**



Source: <https://pixabay.com/en/approved-stamp-stamp-approved-1966715/>
by mstlon | CC0 Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

11


Software Configuration Management



Change Management Process

Factors that should be taken into account when deciding over change requests:

- **The consequence of not making the change**
How serious is the problem?
- **The benefits of the change**
Who will benefit from the change?
- **The number of users affected by the change**
How many users may benefit from the change?
- **The cost of making the change**
Expensive changes are more likely to be rejected
- **The product release cycle**
It may make sense to delay the implementation of the change until the next planned release



Source: <https://pixabay.com/en/cranium-head-human-people-person-3038555/>
by GDU | CC0 Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

12

Software Configuration Management



Change Management for Software Products

- Change management for **software products** (e.g. CAD software) needs to be **handled differently** to that of systems
- Here the **customer is not directly involved**
- **Feedback** comes from the **customer support team**, the company **marketing team**, and the **developers themselves**
- Requests may also reflect **suggestions and feedback of what is offered by competing products**
- **Customers** may use a **webpage** or **email** to report bugs
- Bugs, if valid, are then translated into a formal change request
- **Marketing** may **meet with customers** and **investigate competitive products**
- **System developers** themselves may have some **good ideas**



Source: <https://pixabay.com/en/app-software-contour-settings-1013616/>
by 3dmar_001 (CC0 Creative Commons)

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu

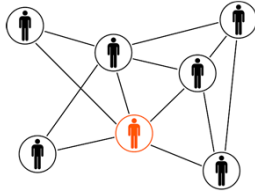
13

Software Configuration Management



Change Management in Agile Methods

- In some **agile methods**, e.g. XP, customers are directly involved in deciding whether a change should be implemented
- When they **propose a change**, they **work with the team to assess** its impact and define its priority
- **Software improvements**, however, are left to the **discretion of the programmers**
- **Refactoring**, is not seen as overhead but as a **necessity**



Source: <https://pixabay.com/en/linked-connected-network-team-153575/>
by OpenClipart-Vectors (CC0 Creative Commons)

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria

Universitätsstraße 15

www.mci.edu


office@mci.edu

14

Software Configuration Management



Change Management in Agile Methods

- Development teams should **maintain a record** of the **changes** made
- This is sometimes called the **derivation history** of a component
- A good way to keep the derivation history is in a **standardized comment at the beginning** of the source code
- You can then write a **simple script** that can **scan all components** and **process the derivation histories** to produce **change reports**
- Change management is usually supported by specialized tools such as for example **Bugzilla** (<http://www.bugzilla.org>)



Source: <https://www.bugzilla.org/>
by www.bugzilla.org / CC0 Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
15

Software Configuration Management



Change Management Derivation History

```
// SICSA project (XEP 6087)
//
// APP-SYSTEM/AUTH/RBAC/USER_ROLE
//
// Object: currentRole
// Author: R. Loeek
// Creation date: 13/11/2009
//
// © St Andrews University 2009
//
// Modification history
// Version  Modifier  Date          Change          Reason
// 1.0      J. Jones  11/11/2009  Add header      Submitted to CM
// 1.1      R. Loeek  13/11/2009  New field       Change req. R07/02
```

Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
16

Software Configuration Management



Version Management

- Version Management (VM) **keeps track of different versions of software** (components)
- It ensures that changes made by different developers do not interfere
- Hence, VM can be thought of the process of managing **codelines and baselines**
- **Codeline:** Sequence of versions of source code
- **Baseline:** Definition of a specific system incl. component versions, libraries and configuration files; they are important because often you have to re-create a specific version of a complete system



Source: <https://pinabay.com/en/code-data-digital-register-13052/>
by PublicDomainPictures | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu

17

Software Configuration Management



Version Management Codeline vs. Baseline

Codeline (A)

A → A1.1 → A1.2 → A1.3

Codeline (B)

B → B1.1 → B1.2 → B1.3

Codeline (C)

C → C1.1 → C1.2 → C1.3

Libraries and external components

L1 L2 Ex1 Ex2

Baseline - V1

A	B1.2	C1.1
L1	L2	Ex1

Baseline - V2

A1.3	B1.2	C1.2
L1	L2	Ex2

Mainline

Source: Sommerville, I. (2011). Software Engineering. 9th Ed. Boston, MA, USA: Pearson Education Inc.

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK


6020 Innsbruck / Austria

Universitätsstraße 15


www.mci.edu

office@mci.edu

18

Software Configuration Management


Version Management Tools




Source: https://upload.wikimedia.org/wikipedia/commons/3/30/Subversion_logo.png
by WidetBot | Creative Commons

- To support version management, you should use **version management tools**
- They **identify, store, and control access** to different versions of components (e.g. CVS, Subversion, Git, Mercurial)

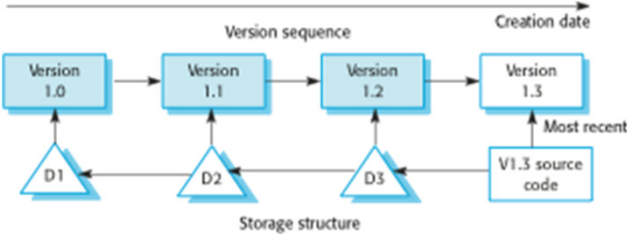
They feature:

- **Version and release identification:** e.g. ButtonManager 1.3 means third version in codeline 1
- **Storage management:** the system stores the difference between versions rather than copies of versions
- **Change history recording:** Describes changes
- **Independent development:** Developers can work on the same component at the same time
- **Project support:** Manage complete components, not only program files

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
19

Software Configuration Management



Version Management Tools



Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
20

Software Configuration Management



Version Management

- Software development is a **team activity**
- Different **team members may work on the same component at the same time**
- Hence version management tools use the concept of **public repositories and a private workspace**
- Developers **check out** components and then **check them in** when finished
- If two or **more people check out the same component at the same time**, the version management **tools will know that and assign separate numbers** to the different check outs
- This may lead to so-called **codeline branches**



Source: <https://pixabay.com/en/matrix-network-data-exchange-1027571/>
by 3dman_eu | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

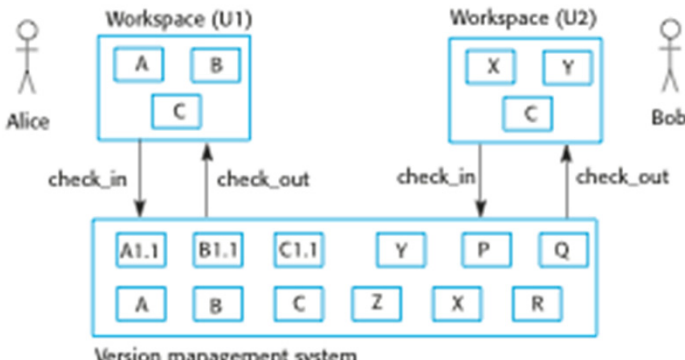
office@mci.edu

21

Software Configuration Management



Version Management Tools – Workspace vs. Repository



Version management system

Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu

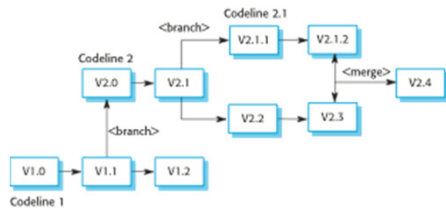
22

Software Configuration Management



Version Management

- At some stage it may be necessary to **merge codeline branches**
- If changes made involve **different parts of the code**, merging can happen **automatically**
- Often, however, there are **overlaps between changes**
- Here a **developer** has to **check for clashes** and may need to modify changes so that they are compatible



Source: Sommerville, I. (2011). Software Engineering, 9th Ed. Boston, MA, USA: Pearson Education Inc.


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

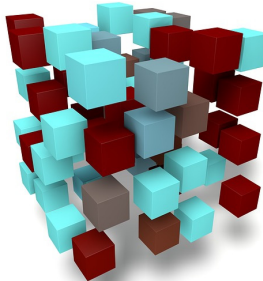
23

Software Configuration Management



System Building

- System building is the process of creating a **complete, executable system by compiling and linking system components, external libraries, configuration files, etc.**
- This involves **checking-out component versions** and using the **configuration description** used to identify the **baseline**



Source: <https://pixabay.com/en/matrix-network-data-exchange-1013612/>
by 3dman_eu | Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

24

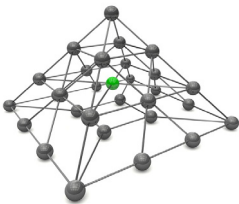
Software Configuration Management



System Building

Building is a **complex, and potentially error-prone process** which may include **various system platforms**

- **The development system**
Including all development tools and programming setups
- **The build server**
Used to build executable versions of the system
- **The target environment**
The platform where the system should execute; often this is a very different system than the development system (e.g. embedded systems)



Source: <https://pixabay.com/en/matrix-network-data-exchange-1013611/>
by 3dman_eu | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria


Universitätsstraße 15

www.mci.edu

office@mci.edu


25

Software Configuration Management



System Building

- System building involves **assembling a large amount of information** about software
- It makes sense to use an **automated build tool**
- Notice that you **not only need the source code** but also **libraries, data files and configuration files**
- Goal is it to **build a complete system** with one **command or mouse click**
- The **build script** includes the **configuration and specification** so that a **build can be generated (compiled) automatically**



Source: <https://pixabay.com/en/programmer-programming-code-work-1453351/>
by JuralMin | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*

MCI MANAGEMENT CENTER INNSBRUCK


6020 Innsbruck / Austria

Universitätsstraße 15

www.mci.edu

office@mci.edu

26


Software Configuration Management


System Building - Agile

- Agile methods recommend very frequent system builds carried out with **automated testing (smoke tests)**
- **These builds are part of the continuous integration strategy**


The steps of continuous integration are:

- Check out mainline system
- Build system and run tests
- Make the changes
- Build system in private space and run tests
- Check in system but do not commit as new baseline
- Build system on build server and run tests
- If the system passes all the tests commit system as a new baseline



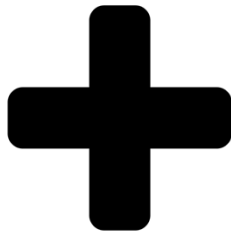
Source: <https://pikabay.com/en/artificial-intelligence-brain-think-3382507/>
by geralt | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
27

Software Configuration Management


System Building – Agile Advantage


- The **argument for continuous integration** is that it allows to **find problems early in the development process**
- It **encourages thorough unit testing**
- Psychologically, **developers are put under pressure** not to 'break' the build
- They **avoid checking in versions of components** that **cause** the whole system to **fail**
- They are therefore **reluctant to deliver** new component versions that have **not been properly tested**
- Hence, **less time is spent** during system testing



Source: <https://pikabay.com/en/sign-plus-black-1923369/>
by Michael Kouassi | Creative Commons

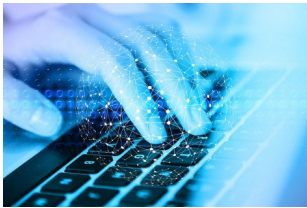
DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
28

Software Configuration Management



Release Management

- A **system release** is a **version of a software system** that is **distributed to customers**
- For mass-market software there are usually two types of releases:
 - Major releases** that add functionality (cost)
 - Minor releases** that fix bugs (free of charge)
- For **custom software** managing system releases is a **complex process**
- **Special releases** may have to be produced **for each customer**
- Hence, a company may need to handle **tens or even hundreds of different releases**
- It may be necessary to **reproduce** exactly the **software** that has been delivered to a customer



Source: <https://pixabay.com/en/network-keyboard-hand-block-chain-3664108/>
by geralt | Creative Commons


DIE UNTERNEHMERISCHE HOCHSCHULE*
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu


29

Software Configuration Management



Release Management

- Therefore, system releases need to be **documented so that they can be re-created**
- Sometimes companies use a **single release of a system for many years**
- To document a release you have to **record the specific version** of the source code components
- You must keep **copies of source code files, executables, data and configuration files** as well as the version of the operating system, libraries, compilers etc.
- For **security updates**, mass market software vendors usually use **automatic updating**
- For **custom systems**, automatic updating usually **does not work**, for the systems are too complex



Source: <https://pixabay.com/en/business-communication-information-2026646/>
by OpenClipart-Vectors | Creative Commons

DIE UNTERNEHMERISCHE HOCHSCHULE*
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

30

Summary

What you should have taken away from this class:

- **Configuration management (CM)** is the **management of an evolving system**. When maintaining a system, CM is put in place to **ensure that changes are incorporated** into the system in a **controlled way**.
- The main configuration management processes are concerned with **change management, version management, system building, and release management**. **Software tools** are available to support all of these processes.
- **Change management** involves **assessing proposals for changes** from system **customers** and other **stakeholders** and deciding if it is cost-effective to implement these in a new version of a system.
- **Version management** involves **keeping track of the different versions of software** components that are created as changes are made to them.
- **System building** is the process of **assembling system components into an executable program** to run on a target computer system.
- Software should be **frequently rebuilt and tested** immediately after a new version has been built. This makes it **easier to detect bugs and problems** that have been introduced since the last build.
- **System releases** include **executable code, data files, configuration files, and documentation**.

References

- Pilato, C. M., Collins-Sussman, B., & Fitzpatrick, B. W. (2004). *Version Control with Subversion*. Sebastopol, CA, USA: O'Reilly Media Inc.
- Vesperman, J. (2003). *Essential CVS*. Sebastopol, CA, USA: O'Reilly and Associates.



MCI[®]
MANAGEMENT CENTER
INNSBRUCK

Thank you for your attention!

mentoring the motivated.

DIE UNTERNEHMERISCHE HOCHSCHULE[®]
MCI MANAGEMENT CENTER INNSBRUCK

6020 Innsbruck / Austria
Universitätsstraße 15

www.mci.edu
office@mci.edu

33