**INTRODUCTION TO VERSION CONTROL**

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

The manual option

One option is to frequently make copies of files that are being frequently modified. This can be done by

copying the files to a directory, CD, flash drive, zip file, or tar file.

This method works, and has been used to manage some fairly complex projects such as the earliest

versions of the Linux kernel. However, this is not a particularly convenient way of managing old

versions of files. This method also becomes increasingly cumbersome as more programmers get

involved in the project.

The lock-modify-unlock option

The first version control systems were central repositories of "good" files. Each software developer

could make modifications to a working copy in their own account then put those changes into the

repository. The problem of two people modifying the same file at once was solved by a lock-modify-

unlock system. When a developer wanted to modify a file, they checked it out which put a lock on the

file that prevented anyone else from editing the file. When they were done, they checked their new

version into the central repository and the lock was removed.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

The manual option

One option is to frequently make copies of files that are being frequently modified. This can be done by

copying the files to a directory, CD, flash drive, zip file, or tar file.

This method works, and has been used to manage some fairly complex projects such as the earliest

versions of the Linux kernel. However, this is not a particularly convenient way of managing old

versions of files. This method also becomes increasingly cumbersome as more programmers get

involved in the project.

The lock-modify-unlock option

The first version control systems were central repositories of "good" files. Each software developer

could make modifications to a working copy in their own account then put those changes into the

repository. The problem of two people modifying the same file at once was solved by a lock-modify-

unlock system. When a developer wanted to modify a file, they checked it out which put a lock on the

file that prevented anyone else from editing the file. When they were done, they checked their new

version into the central repository and the lock was removed.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

The manual option

One option is to frequently make copies of files that are being frequently modified. This can be done by

copying the files to a directory, CD, flash drive, zip file, or tar file.

This method works, and has been used to manage some fairly complex projects such as the earliest

versions of the Linux kernel. However, this is not a particularly convenient way of managing old

versions of files. This method also becomes increasingly cumbersome as more programmers get

involved in the project.

The lock-modify-unlock option

The first version control systems were central repositories of "good" files. Each software developer

could make modifications to a working copy in their own account then put those changes into the

repository. The problem of two people modifying the same file at once was solved by a lock-modify-

unlock system. When a developer wanted to modify a file, they checked it out which put a lock on the

file that prevented anyone else from editing the file. When they were done, they checked their new

version into the central repository and the lock was removed.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

Why have version control?

Anyone who has put a large amount of work into software development would be upset if the source

code files were lost due to a hardware failure or incorrectly typed command. Some computers have an

automatic nightly backup, and others don't. Even when there is a nightly backup, it would be convenient

to be able to go back to the version of a file as it looked a few hours ago when some new code didn't

work as expected. This is a particularly important issue in software development because a given file

may be modified many, many times as the software is written. Version control systems are designed to

store, compare, and manage many versions of computer software source code files.

The manual option

One option is to frequently make copies of files that are being frequently modified. This can be done by

copying the files to a directory, CD, flash drive, zip file, or tar file.

This method works, and has been used to manage some fairly complex projects such as the earliest

versions of the Linux kernel. However, this is not a particularly convenient way of managing old

versions of files. This method also becomes increasingly cumbersome as more programmers get

involved in the project

**What is version control?**

Three basic principles:

1. Keep a record and history of changes.

2. Give public access to the information.

3. To maintain different versions from the same data set.

**What types of data?**

Source code,

Documentation

Configuration files

Generally, any type of data

**What is Version Control?**

Terminology

Repository

A centralized copy of all files being tracked, structured in directory trees

**Working copy;**

A local copy of data that can be changed and synchronized with the repository. Contains special information that allows for synchronization.

**Version;**

A group of directories and files that reflect the repository state at a determined moment. Terminology Basics The repository is the master copy All work is done in a work copy Changes are reflected and appear in the repository (using the commit command)