Study Notes for Comp 4350 - Software Engineering II

Version Control

- **Definition**: The practice of tracking and managing changes to software code.
- **Systems**:
  - **Software Tools**: Help teams manage source code changes over time.
  - **Types of VCS**:
    - **Centralized Systems**: Single master repository (e.g., Subversion).
    - **Distributed Systems**: Multiple repositories (e.g., Git).
- **Version Control Functions**:
  - Track complete change history of files.
  - Support branching and merging for independent workstreams.
  - Provide traceability connecting changes to bug tracking and project management.
- **Key Concepts**:
  - **Codelines**: Sequences of source code versions.
  - **Baselines**: Specific system definitions, including component and configuration versions.
  - **Dependencies Graph**: Visual representation of component interdependencies.

Software Configuration Management (SCM)

- **Purpose**: Manage and track changes in software configuration.
- **Components**:
  - Code Management
  - Dependency Management
  - Environment Configuration (Software, Hardware, Infrastructure)
- **Configuration Types**:
  - Build-time
  - Deploy-time
  - Runtime
- **Strategies for Managing External Libraries**:
  - Keep locally
  - Download from the internet
  - Build local external libraries repository (e.g., Maven for Java)

Coding Practice

- **Understandability**: Code should be written with human readers in mind.
- **Tips for Writing Understandable Code**:
  - Use meaningful identifier names
  - Add comprehensive comments that complement the code
- **Research on Automated Comment Generating**:
  - Sequence-to-sequence (Seq2Seq) model: Convert sequences from one domain to another.
  - Utilization of deep neural networks for comment generation algorithms.

Pair Programming with AI

- Discussions about enhancing coding practices by pairing with AI technologies.

Coding Conventions

- Defined as a set of guidelines for specific programming languages.
- Recommend programming style, practices, and methods.
- **Resources**:
  - Google Style Guides
  - NASA C Programming Guidelines

Software Change Models

- **Tree Evolution Model**: New software configuration versions replace older ones.
- **Linear Evolution Model**: One new version directly replaces the old version.

Configuration Management in Other Domains

- Used in military systems, civil engineering, and other industrial engineering segments.

References for Further Reading

- *Continuous Delivery* - Jez Humble & David Farley (Chapter 2: Configuration Management)
- *Software Engineering* - Ian Sommerville, 10th Edition (Chapter 25: Configuration Management)

Tips for Open Book Exam Preparation

- Ensure quick access to these sections through bookmarks or indexing.
- Understand the workflow and basic commands of both centralized and distributed VCS.
- Review different maintenance and evolution models of SCM.
- Familiarize with coding practices and industry standards.
- Identify examples in varied contexts, like SCM in civil engineering.
- Be able to compare and contrast VCS types (centralized vs distributed).

Class Notes

Slide: 1

Title: Comp 4350 Software Engineering II Dr. Shaowei Wang

Slide: 2

Title: Administrative item Build team as soon as possible

Slide: 3

Title: Agenda Version control Coding practice

Slide: 4

Title: Software systems are constantly changing New versions are created to handle software changes Each version may have to be maintained and managed

Slide: 5

Title: Multi-version systems For large systems, there is never just one ‘working’ version of a system. There are always several versions of the system at different stages of development. Different versions of the system may be in use

Slide: 6

Title: Multi-version system development Development system

Slide: 7

Title: Versions of printer firmware products - a tree evolution model In the tree evolution model, one or more new software configuration versions will replace former software configuration version. Even the same component needs different configuration to be maintained and tracked.

Slide: 8

Title: Version control Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time.

Slide: 9

Software item (SCI) Design documents Software code Data files including files of test cases and test scripts Software development tools

Slide: 10

No content

Slide: 11

Deployment code, build code Dataset for machina-learning systems.

Slide: 12

Title: Why version control? A complete long-term change history of every file Branching and merging Traceability

Slide: 13

Help traceability in software project

Slide: 14

Code

Slide: 15

Code

Slide: 16

Codelines and baselines

Slide: 17

Codelines and baselines

Slide: 18

Codelines and baselines Baselines representing different versions of a system

Slide: 19

No content

Slide: 20

Versions of printer firmware products - a tree evolution model Baseline Codeline

Slide: 21

Version control Code Dependencies Software configuration Environment

Slide: 22

Dependencies graph for browserify in NPM https://npm.anvaka.com/#/view/2d/browserify

Slide: 23

No content

Slide: 24

Manage external libraries Keep locally Download from the internet Build local external libraries repo in the company

Slide: 25

No content

Slide: 26

Version control Code Dependencies Software configuration Environment

Slide: 27

Software configuration management Configuration information can be used to change the behavior of software at build time, deploy time, and runtime.

Slide: 28

MySQL configuration parameters

Slide: 29

Change log level during runtime in Spring Boot

Slide: 30

Flexibility usually comes at a cost!

Slide: 31

Type of configuration

Slide: 32

Version control Code Dependencies Software configuration Environment

Slide: 33

Environment Software Hardware Infrastructure

Slide: 34

No content

Slide: 35

Types of environment configuration

Slide: 36

Key principle - make their creation a fully automated process. Reduce risk Replicability

Slide: 37

No content

Slide: 38

Version control systems Centralized systems Distributed systems

Slide: 39

Key features of version control systems Version and release identification Change history recording Support for independent development Project support Storage management

Slide: 40

Benefits of distributed version control

Slide: 41

Open source development

Slide: 42

Storage management Delta-based approach

Slide: 43

Storage management using deltas

Slide: 44

Storage management in Git

Slide: 45

Tips

Slide: 46

Semantic versioning of software releases

Slide: 47

Semantic versioning of software releases

Slide: 48

Best practices for versioning a release

Slide: 49

Activity1: Introduce to each other

Slide: 50

Agenda Version control Coding practice

Slide: 51

Understandability: What to keep in mind?

Slide: 52

What does this code do?

Slide: 53-64

Content related to code understandability, comments, and code generation research

Slide: 65-67

Content related to coding conventions and understandability

Slide: 68

No content

Slide: 69

References

Slide: 70

Versions of an accounting software package - a linear evolution model

Slide: 71

Configuration management