# Lecture 15:
## Configuration Management, Software Maintenance, and Code Reviews (Part I)

Valentin Razmov

---

## Outline

- Configuration management
  - Motivation
  - Challenges
  - Examples
  - Reality

---

## Resources

- *The Pragmatic Programmer*, by Andrew Hunt and David Thomas
- *Code Complete*, by Steve McConnell
- *Rapid Development*, by Steve McConnell

---

## Configuration Management – What and Why?

- **What:** A technique for simultaneously handling multiple versions (configurations) of a set of documents (code) by multiple people
- **Why:** (List the two main reasons that you see.)

---

## Reasons for Doing Configuration Management

- Allows multiple people to collaborate in producing a common piece of work without imposing elaborate or overly strict management rules
  - E.g.: "You can only touch the code between 8am and 10am, then it's my turn."
- Reverting to a previous version of the code
  - Undoing undesirable changes
  - Allowing testers to work independently on a different (not latest) release

---

## Reasons for Doing Configuration Management (cont.)

- Tracking changes and project progress metrics
  - Who makes the largest number of changes?
  - How large/extensive are individual changes?
  - Who changes this line of code?
  - Which modules get changed most often?
  - How much did module X change since the last release?
    - In lines of code (LOC), new methods, etc.

## Reasons for Doing Configuration Management (cont.)

- Splitting source tree into multiple versions / branches and later merging (some of) them
  - Motivation: different customers need / pay for different updates
  - Challenge: avoid duplication of effort across different branches of the tree
- Central place / repository that keeps an authoritative version of the work
  - Can be properly protected and managed with fewer human resources
  - Everyone knows where to look for things
  - Can run (repeatable) automatic builds & regression tests

## High-level Motivation for Configuration Management

- Maximizing effectiveness of the team
  - Efficient collaboration on a common body of work
  - People can work on snapshots of the system, created at different points in time
- Robustness (against mistakes)
- Accountability (of contributors)
- Managing complexity (and avoiding nightmares)

## The Hardest Challenges with Using CMS

- A human has to keep "in mind" all the different configurations and decide which updates to apply to which branches, and when
  - Gets increasingly error-prone with large projects and many branches (releases, customers)

- Challenges you've experienced?

## Examples of Configuration Management Systems (CMS)

- CVS / RCS
- Perforce
- Subversion
- Visual Source Safe
- Roundtable
- others

Note: Not all CMS are created equal.

## In Reality...

- Most companies use some CMS
  - Including all companies where I have worked
- I have heard of companies that don't use a CMS
  - It is a huge risk for none / dubious cost savings
  - Management can easily buy into using a CMS to reduce risks

## Relevance of CMS

- Applicable to all types of valuable documents, not just source code
  - "You don't need to floss all your teeth – just the ones you want to keep."