Software Configuration Management

Slides derived from Dr. Sara Stoecklin’s notes and various web sources.
What is SCM?
SCM goals

• Manage the changes to documents, programs, files, etc.
• Track history
• Identify person responsible for each change, and reasons
• Recover (roll back to) previous versions as necessary
• Maintain sets of compatible versions of files (configurations)
Several related concepts & terms

- **Source code vs. configuration management**
  - both called SCM
- Revision control ≈ version control
- Web content management
- Work flows
Definition according to Wiki

Configuration management is the management of features and assurances through control of changes made to **hardware**, **software**, **firmware**, **documentation**, test, test fixtures and test documentation of a system, throughout the development and operational life of a system.

*Source Code Management* or **revision control** is part of this.
Definition According to Dennis

Configuration and change management is the tracking of the state of the artifacts (hardware, software, firmware, documentation, test, test fixtures and test documentation of an system) throughout the development of a system.
Configuration Management Defined

The management and control of all any changes made to any and all features of the software development activity. This includes hardware, software, documentation, and firmware.

Configuration management can be done with a tool. These tools range in price from $0 to $400,000. It depends on how many features you wish to manage and how well you want to manage them.
Why do we care?

*Remember Apollo 13.*

If a change needs to be made to ANY artifact then when you put them all together again you had better make it right. Suppose you have a programming error and need to change it, recompile and use that software.

What version of the hardware did you use for the original? What version of the OS, DB, compiler? What version of the file management software? All that should be the same to recompile. So we must manage.
Why do we care?

Suppose we change requirements and get a new use case diagram. Do other things have to change? Will we have more actors, attributes, screen? So even when the documentation of the system changes as it evolves, we must track that, too.
**What are the artifacts?**

**Hardware** - ANY hardware that is used by the system.

**Software** - system software (OS, DB, Compiler, etc), supporting software (sorters, mergers, utilities), application software (you wrote or you use).

**Firmware** - ANY firmware used by the system.

**Documentation** - deliverables for development, documentation maintained for the operation of the system, etc.
Configuration Management Includes

**Change** management - of changes to the specifications of a potential software system

**Documentation** management - of all documentation including defects, specification, testing, purchasing, emails, memos, agendas...every single documentation detail

**Hardware/firmware** configuration management - of all the hardware and firmware.

**Source code** management - of changes to the source code including the application code, operating system, compilers, utilities, database management system, communication system, middleware, etc.
Configuration Management Standards

e.g.


GEIA Standard 836-2002 Configuration Management Data Exchange and Interoperability
Source Code Management

• One of the important parts of configuration management
• For any development project you need to have a SCM plan
• SCM is closely connected with team workflow
  – Workflow defines how SCM is done
  – How tools are used
Complicating factors for SCM

- Multiple versions
- Multiple branches (e.g., development, bug fix, old releases)
- Multiple authors
- Concurrent activities
- Geographical distribution
- Disk crashes, human errors
SCM Model Differences

• Centralized vs. Distributed
• Pushed vs. pulled updates
• Handling of concurrent updates
  – Atomic commit operations (transactions)
  – File locking
  – Version merging
There are many SCM tools

- RCS
- CVS
- Subversion (svn)
- Git
- Mercurial (hg)
- Bazaar (bzr)
- ... many more
Key terms and concepts

- File vs. configuration
- Version/revision numbers
- Timestamps
- Releases
- Repository
- Working copy, or sandbox
- Baseline/trunk/mainline/master
- Branches/forks
- Change list or patch
More terms and concepts

- “Checkout” has multiple meanings
  - Get local working copy (or make it visible?)
  - Locked, or not?
  - Configuration versus file
- Export/import
- Commit
- Conflict (superficial, or deep)
- Merge
- Tags
Centralized vs. distributed SCM models
Centralized SCM

Operations require server
- single point of failure
- bottleneck

Decentralized SCM

Anyone can be a server

How decentralized SCM works.
Start with a global repository upstream
Clone it

upstream

local
Can make cheap local clones via links

Changes can be **pushed back** upstream

published to web

or shared with trusted peers

Benefits of decentralization

- Non-intrusive micro-commits
- Detached operation (off net)
- No single point of failure
- Backups are trivial
- Very flexible
- ... ?
Problems with decentralization

- No “locking”
- No single authoritative version
- Relies on clock synchronization
- Requires greater discipline, imposed by team workflow rules
Example workflow models

Suitable for use with Git or other distributed SCM tool.
Agile workflow

All developers are equal, all push changes to same repository. Repository is always up to date with the current “wave front”.

UML Sequence Diagram for an Agile team.
Agile workflow
Managed repository

Only one person can push changes to the blessed repository.
Means less chance for accidents, but we now have a bottleneck.

http://progit.org/book/ch5-1.htm
An alternate way of sharing a repository, using branches.

http://agentdero.cachefly.net/unethicalblogger.com/images/basic_slide_workflow.png
Linux-like hierarchical model

Android Git Workflow

http://source.android.com/submit-patches/workflow

Two kinds of review, verification by multiple testers before admission to repository.
In this course

• If you use an IDE such as VB Studio or Eclipse then you can use a SCM plug-in
  – Several are available.
• However, if you are not using an IDE you will need to use command-line operations, scripts, or a separate GUI tool for version control.
• For Spring 2010 we will use Git.
  – So, you will need to learn it.
  – During your career you will probably need to learn and use several others.