# Notes from activities:

Activity 1.

1. Configuration Items :

* A document, drawing, software element or hardware unit whose development is to be controlled and tracked.
* Items have a development history so that any item may be reconstructed in one of its former states.

1. Software Configuration Management

* Simultaneous Update
  + When two or more programmers work separately on the same program, the last one to make the changes can easily destroy the other's work.
* Shared Code
  + Often, when a bug is fixed in code[,](http://www.cmcrossroads.com/cgi-bin/cmwiki/edit/CM/Code?topicparent=CM.SoftwareConfigurationManagement) shared by several programmers, some of them are not notified.
* Common Code
  + In large systems, when common program functions are modified, all the users need to know. Without effective code management, here is no way to be sure of finding and alerting every user.
* Versions
  + Most large programs are developed in evolutionary releases. With one release in customer use, another in test, and a third in development, bug fixes must be propagated between them. If found by a customer, for example, a bug should be fixed in all later versions. Similarly, if a bug is found in a development release, it should be fixed in all those prior versions that contained it. In larger systems with several simultaneous active releases and many programmers working on bug fixes and enhancements, conflicts and confusion are likely.
* AccuRev SCM - AccuRev
* Auto-trol Technology - KONFIG CM
* BitMover, Inc - BitKeeper
* Confignium - Versium SCM
* TeamForgeSCM/CloudForge/Subversion- CollabNet
* CVS Suite - March Hare Software
* Git – Open source (by Linux Kernel developers)
* Mercurial – Matt Mackall under GNU GPL

Activity 2.

Overview of Open Source Software development

* hosting for thousands of Open Source Software projects.
* free to download and use
* encourage you to be in the Open Source Software community.

# Users of these site – two types:

1. Consumers – download software
2. Developers – contributors:

* Donate
* Report bugs
* Create documentation
* Join a project
* Submit patch files
* Revive an abandoned project
* Create a new project

Activity 3.

1. Vertical team organization

A vertical team is composed of generalists. Use cases are assigned to individuals or small groups, who then proceed to implement the use case end to end.

Advantages

* You have smooth end-to-end development on an individual use case basis.
* Developers gain a wider range of skills.

Disadvantages

* Generalists are typically high-paid consultants who are difficult to find.
* Generalists typically do not have the specific technical expertise required to quickly solve detailed problems.
* Subject matter experts may have to work with several groups of developers, increasing their burden.
* All generalists are not created equal.

Success factors

* Everyone is working to a common set of standards and guidelines.
* Good communication between developers is required to avoid common functionality being implemented by various teams.
* Common, and agreed to, architecture needs to be developed early in the project.

1. Horizontal team organization

A horizontal team is composed of specialists. This team works on several use cases simultaneously, each member working on their own aspects of the use case.

Advantages

* A higher quality of work is performed for each aspect (requirements, design, and so on) of the project.
* External groups, such as users and operations staff, interact with a small group of specialists who understand their exact needs.

Disadvantages

* Specialists often do not appreciate the importance of other specialties, resulting in disconnects between various aspects of the project.
* Information required by "back-end" people may not be gathered by the "front-end" people.
* Project management is more difficult because of competing priorities, visions, and needs of specialists.

Success factors

* Good communication is required between team members so that they understand where each person is coming from.
* Defined processes and quality gates that specialists must follow to promote effective hand-off to other specialists are required.

1. Hybrid team organization

A hybrid team is made up of both generalists and specialists. The generalists stay with a use case throughout its development, supporting and working with specialists who work on portions of several use cases.

Advantages

* You get the best of both worlds.
* External groups interact with a small group of experts.
* Specialists focus on what they are good at.
* Individual use cases are implemented consistently.

Disadvantages

* You get the worst of both worlds.
* Generalists are still hard to obtain.
* Specialists still may not appreciate and work well with other specialists, although this should be tempered by the generalists.
* Project management is still difficult.

Success factors

* Good team communication is required.
* Common architecture needs to be developed.
* Common processes, standards, and guidelines must be well defined.