

# Introduction to CM

Dependable Software Laboratory  
정세진

2016. 03. 04

---

# Contents

- Configuration?
- Configuration Management
- Necessity
- Configuration Activities
- Tools
- Term Project with CM

---

# Configuration?

- Configuration = 형상
- 소프트웨어 개발 산출물 (source code, document 등) 이 배치되어 있는 배열

## **configuration** 영어

미국/영국 [kənfigjʊreɪʃən]

① 구성   ② 형상   ③ 상대적 배치   [더보기](#)

---

---

# Configuration Management

- Configuration Management (CM)
  - 형상 항목을 식별하여 그 기능적 물리적 특성을 문서화하고, 그러한 특성에 대한 변경을 제어하고, 변경 처리 상태를 기록 및 보고하고, 명시된 요구사항에 부합하는지 확인하는 기술적이고 관리적인 감독, 감시 활동 [IEEE-Std-1042]
  - The process of managing changes to an evolving **software**
    - The aim of CM is to support the system development and integration process
    - CM is not limited in source code
  - SCM manages all software entities
    - Management plan, Spec, User documents, Test data, Source code, Libraries, Etc. [IEEE-Std-1042]

# Configuration Management

- Configuration Management (CM)
  - 형상 항목을 식별하여 그 기능적 들  
변경을 제어하고, 변경 처리 상태를  
는지 확인하는 기술적이고 관리적
  - The process of managing changes:
    - The aim of CM is to support the s
    - CM is not limited in source code
  - SCM manages all software entities:
    - Management plan, Spec, User doc  
[IEEE-Std-1042]

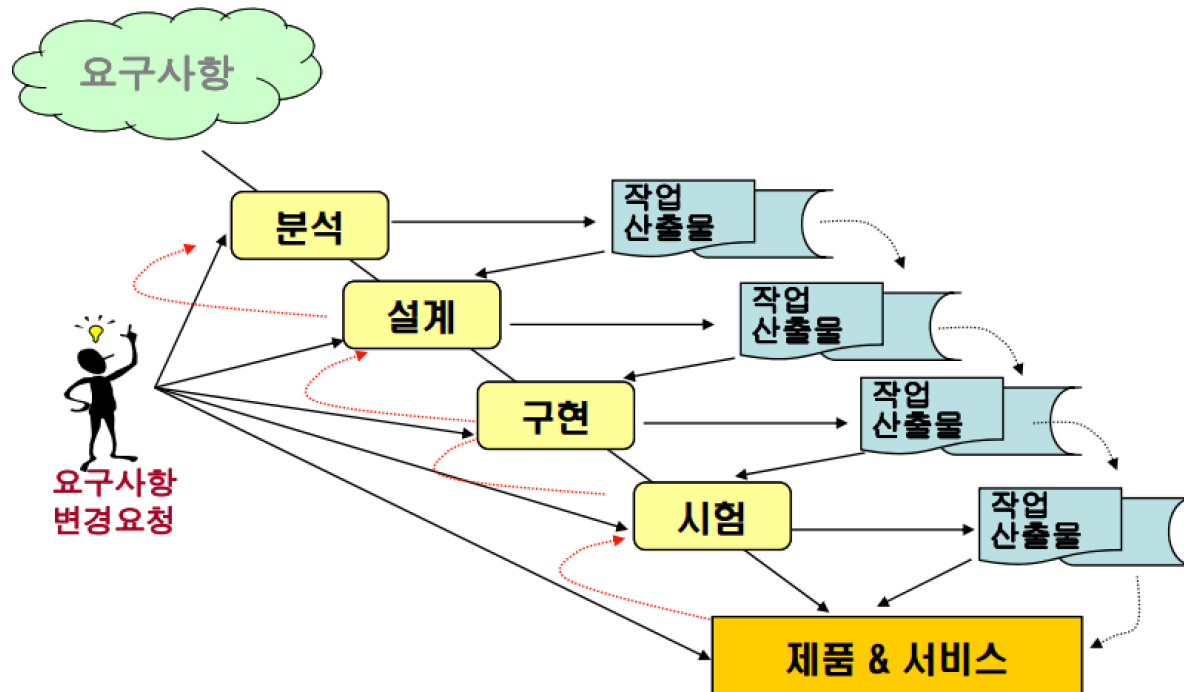
**2.1.3 SCM Manages all Software Entities.**  
Software CM extends the management disciplines of hardware CM to include all of the entities of the product as well as their various representations in documentation. Examples of entities managed in the software engineering process include

- (1) Management plans
- (2) Specifications (requirements, design)
- (3) User documentation
- (4) Test design, case and procedure specifications
- (5) Test data and test generation procedures
- (6) Support software
- (7) Data dictionaries and various cross-references
- (8) Source code (on machine-readable media)
- (9) Executable code (the run-time system)
- (10) Libraries
- (11) Data bases:
  - (a) Data which are processed,
  - (b) Data which are part of a program
- (12) Maintenance documentation (listings, detail design descriptions, etc)

한  
하

# Necessity


- Requirements can always be changed by client, manager, etc.





---


# Necessity


- Potential problems in project
  - Requirements are always be changed by several reasons
  - The results of modifying product(e.g. source code, document) can not be notified to manager
  - Several developers are able to do duplicated work
  - It may exist one version with several copies
- Minimize the risk in accordance with the results about changing of the requirements and characteristics of the software

 Requirements\_rev1

 Requirements\_rev2

 Requirements\_rev3

 Requirements\_최종

 Requirements\_최종\_1

# Necessity

## A사의 사례

- 각자의 진도에 따라 개발
  - 개발 프로세스의 효율성 저하
- 작업PC와 네트워크에 수많은 중복된 파일이 존재하거나 어디 있는지 모름
  - 소프트웨어 자산 관리의 체계화와 보호 미흡
- 변경에 따른 비용과 위험의 증가, 산출물의 형식이 개인마다 다름
  - 통합관리와 품질향상의 저해요인

## B사의 사례

- 동일한 문제 재발
  - 개발 S/W의 중복보관에 의한 버전 불일치
  - 병렬 개발 환경에서의 S/W 개발 시 동시 수정에 의한 관리의 어려움
- 개발자 퇴사 시 자료 관리, 개발이력 파악의 어려움
  - 개발 S/W의 산재로 인한 관리, 통제, 공유의 어려움
  - 관련된 자료의 버전 및 이력 관리의 어려움
  - 신입/경력 연구원의 개발 참여 시 기존 개발 History 파악이 어려움



# Configuration Management about COTS SW in NPP

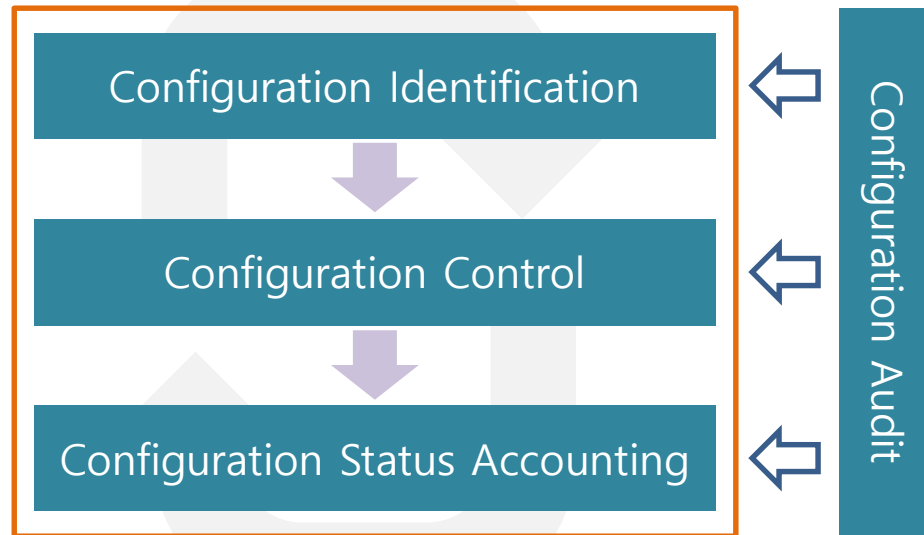
Table A-4. Software Configuration Management Criteria

1	Does the configuration management plan cover the minimum required subjects in the required format?	Format and subject matter is standard-dependent, but most standards specify similar approaches. See IEEE 828
2	Does the plan describe responsibilities, authority, and relations between configuration management units and software development units?	IEEE 828
3	At least one configuration control board (CCB) is required. Does the plan describe the duties and responsibilities of the CCB and relations between the CCB, SQA, and software developers? e.g., Authority & responsibility Role Personnel How appointed Relation of developers & users	IEEE 828
4	Does the configuration management operation provide the following required functions? Configuration ID (baselines) Configuration control Configuration status accounting & reporting Configuration audits & reviews	IEEE 828
5	Configuration management is founded upon the establishment of "configuration baselines" for each version of each product. Is each product or version uniquely identified and "baselined"?	IEEE 828
6	Is the level of authority required for change (i.e., change control) described? Appropriate subjects include: Change approval routing lists Library control Access control R/w protection Member protection Member identification Archive maintenance Change history Disaster recovery Authority of each CCB over listed configuration items	IEEE 828
7	Does status accounting include Data collection Identified reports Problem investigation authority Maintaining and reporting Status of specifications Status of changes Status of product versions Status of software updates Status of client-furnished items	IEEE 828

8	Are suppliers of software products (e.g., COTS) under control? For each supplier. . . Is the SCM capability known? How is SCM performance monitored? For each product... Is the version in use archived? Is the version ID'd & baselined? Is the product under change control? Are product interfaces under control? Are suppliers CM audits "visible"? Is there valid problem tracking? Regarding supplier records. . . What records are kept? Can reviewers obtain access to them? How good are they? What security does the supplier have?	IEEE 828 and IEEE 1042. ISO 9000-3
9	Are the records to be maintained identified and are there retention periods specified for each type of record?	IEEE 828
10	What additional policies and directives govern the configuration management?	See Table A-14 for a list of typical policies and directives.

# Configuration Activities

- Configuration Identification
  - 형상 관리 항목 식별
- Configuration Control
  - 변경 요청 시 변경 여부 및 활동 통제
- Configuration Status Accounting
  - 변경 내용 기록 및 보고
- Configuration Audit
  - 형상 항목 변경에 대한 확인 및 감사



---

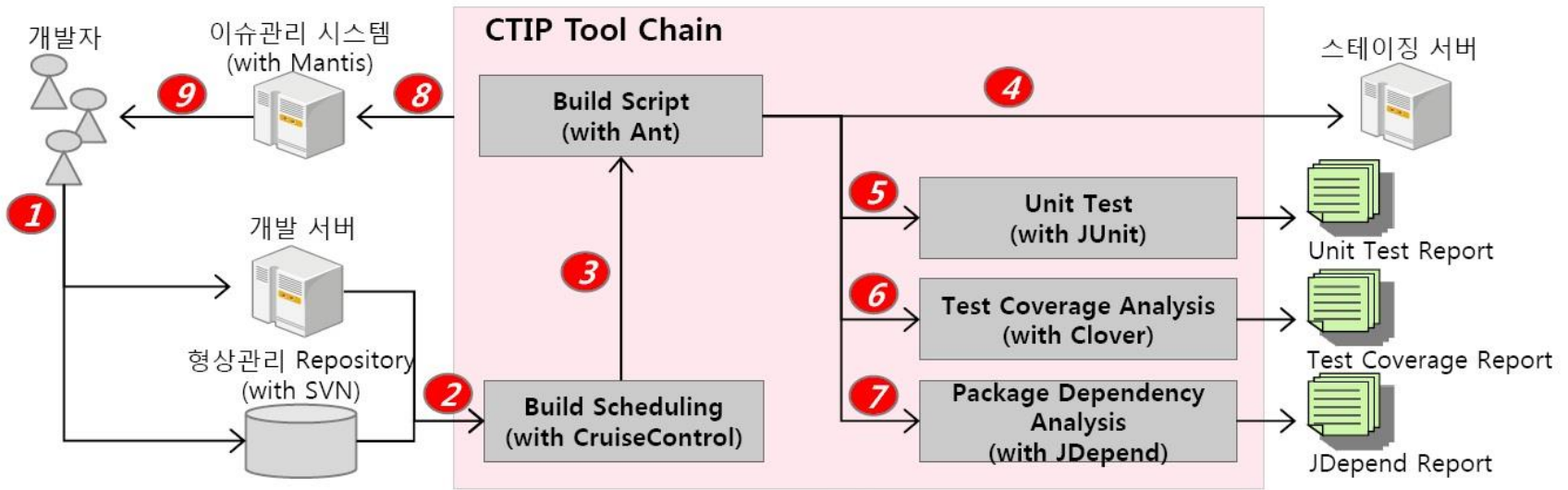
# Tools

- **Tools for supporting CM activities**

- 소스코드 또는 문서의 버전, 이력 등 변경사항을 체계적으로 관리 할 수 있는 기능 제공
    - 형상관리 도구
    - 프로젝트 관리 도구 등
  - CVS : Version Management
  - ClearCase : Integrated CM Tool
  - git
  - subversion(svn) : Version management
  - Mantis : issue, bug tracking
  - Etc.
- 
- 참조 : <https://www.swbank.kr/helper/tool/toolMain.do>

# Term Project with CM

CTIP(Continuous Test & Integration Platform) 환경 구성 및 개발 흐름도



- 1 소스코드 개발 및 관리
- 2 최신 소스코드 자동 체크아웃(업데이트)
- 3 정해진 스케줄링에 따라 빌드 요청
- 4 빌드 자동 수행 및 배포
- 5 빌드 시 단위 테스트 자동 수행
- 6 빌드 시 커버리지 분석 자동 수행
- 7 빌드 시 의존성 분석 자동 수행
- 8 빌드 후 문제점 확인 및 등록 (통합 담당자)
- 9 등록된 오류 확인 및 해결 (개발자)

---

# Term Project with CM

## 팀 프로젝트 주제 : Clone Checker

### - 개발내용:

- 다수의 C 프로그램을 대상으로 상호 cheating 여부를 정량적으로 판단하고, 해당 내용을 OOO 하게 알려주는 프로그램을 개발
- 다양하고 정교한 cheating cases를 발견할 수 있도록 알고리즘을 잘 개발하고 설계

### - 개발환경: JAVA / GUI / Eclipse

### - 도구 사용

- UML, SVN, JUnit, JFeature, Mantis 및 각종 CTIP 환경
- NIPA 소프트웨어뱅크 ([소프트웨어 개발도구 소개](#))

### - 2학년 "[프로그래밍 프로젝트](#)" 수업과 연동

- 2학년 수업에서 개인 및 팀프로젝트로 개발한 각종 프로그램들의 Test Cases로 사용하여, Cheating 여부를 확인합니다.

### - 4학년 "[소프트웨어 검증](#)" 수업과 연동

- 4학년 수업에서 수행한 시스템 테스트 및 정적분석 결과를 반영하여 다음 cycle을 진행합니다. ([시스템테스트대응서\(STR\)](#) 및 [정적분석대응서\(SAR\)](#) 제출)
- CTIP 환경을 전수 받아 설치한 후, 4학년 학생팀과, [SVN](#)을 사용하여 코드를 공유하고, [Mantis](#)를 이용하여 이슈를 관리합니다.
- CTIP 환경에서 [JUnit](#)을 사용하여 단위시험을 수행한 후 [단위시험보고서\(단위시험\)](#)를 제출 및 발표합니다.
- 4학년 수업에서 수행한 [정적분석보고서](#)를 결과를 반영하여 [정적분석대응서](#)를 제출 및 발표합니다.

# Term Project with CM

- 4학년 "소프트웨어 검증" 수업과 연계
  - CTIP (Continuous Test & Integration Platform) 환경 제공
    - 자동 build, testing, 정적 분석 등
  - **SVN 을 이용한 버전 관리**
  - Mantis 를 이용한 이슈 관리

보고서 #2	팀발표 #1	팀발표 #2	팀발표 #3	팀발표 #4
SVN, Mantis, JFeature, JUnit 사용법 및 CTIP 개론	OSP Stage 1000	OSP Stage 2030	OSP Stage 2040	OSP Stage 2050/2060
	Planning	Analysis	Design	Implementation & Unit Test
보고서	발표자료 보고서 wmv	발표자료 보고서 wmv	발표자료 보고서 wmv	발표자료 소스코드 설치 단위시험보고서 시스템시험보고서 wmv 발표자료

---

# 담당 조교

- 담당 조교

- 정세진 (DSLlab, 새천년관 1006호)

- E-mail : jsjj0728@gmail.com, jsjj0728@konkuk.ac.kr

- [SMA2016] 000~~