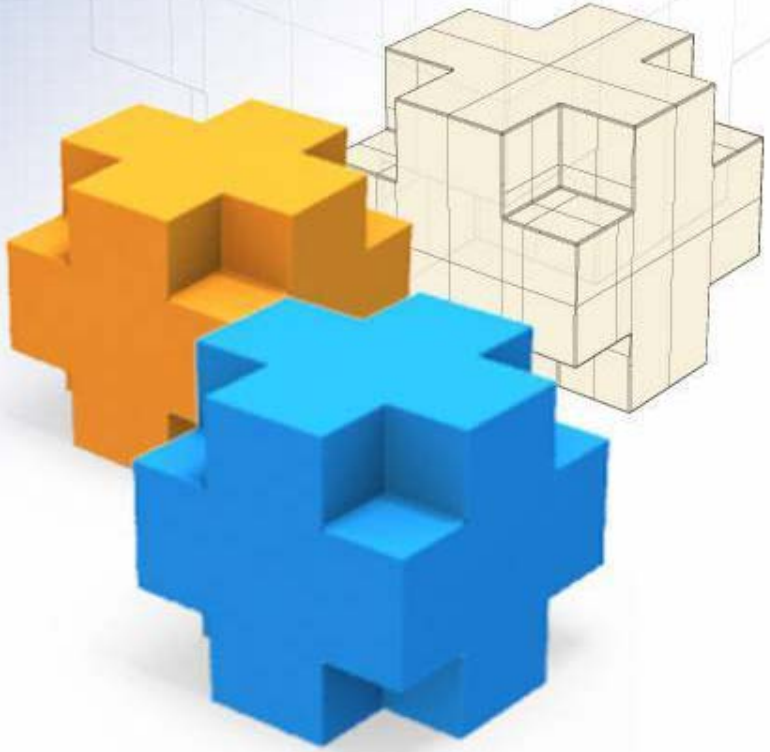
LECTURE 17

품질보증



일정(01강좌)

- 14주(12/1, 3) – 품질
- 15주(12/8) – 프로젝트 데모(각 팀 당 15분 발표)
  - 야구 작전 분석시스템
  - 미아 찾기 시스템
  - Knight Guard
  - 우리 딸 어디니?
  - 시간표 작성 시스템
  - Auto Time-table Maker
- 15주(12/10) – 프로젝트 데모 (각 팀 당 15분 발표)
  - 미래형 냉장고
  - 친구 찾기
  - Food Manager
  - PC 업그레이드 관리
  - 친구 만들기
  - Smart Shopping Map

일정(02강좌)

- 14주(12/1, 3) – 품질
- 15주(12/8) – 프로젝트 데모(각 팀 당 20분 발표)
  - 휘트니스 클럽
  - 탐색기에 날개 달자
  - 뒤풀이 플래너
  - 지하철 승하차 안내
- 15주(12/10) – 프로젝트 데모 (각 팀 당 20분 발표)
  - 동국로드뷰
  - 도서관리시스템
  - 울트라 메모리
  - Search Cube

품질 개념

❑ 내용

- ❍ 품질의 의미
- ❍ 품질 요소
- ❍ 소프트웨어 특성과 품질
- ❍ 프로세스 품질
- ❍ 프로덕트 품질

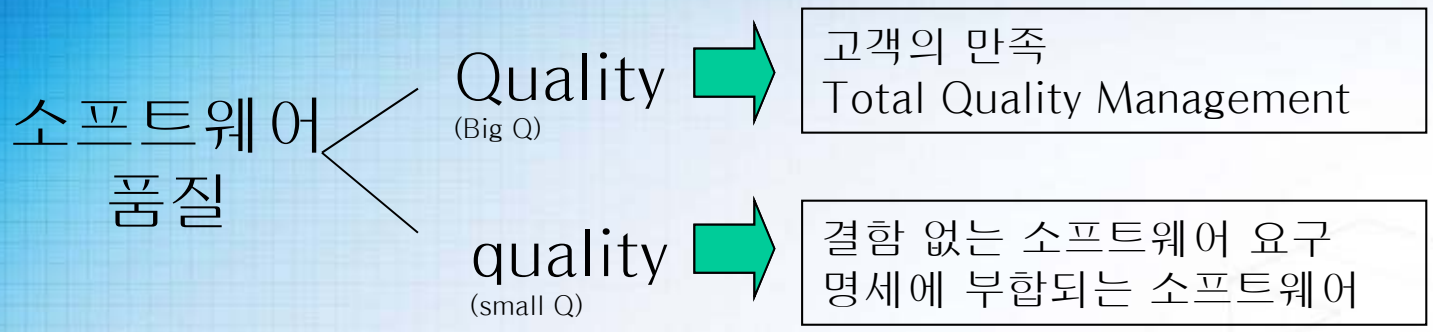
품질 개념

- 좋은 품질의 소프트웨어란?
  - 사용자 관점: 원하는 기능을 제공하여 유용하고 쉽게 쓸 수 있고 신뢰성 높고 필요에 따라 계속 발전하는 소프트웨어
  - 개발자 관점: 좋은 설계 구조를 가지며 쉽게 유지 보수할 수 있고 테스트가 용이하고 환경에 맞도록 변경 가능한 소프트웨어

- 품질 측정 가시화의 어려움
  - 구체적으로 원하는 수준의 품질인지 판단
  - 다른 소프트웨어와의 품질 비교

품질의 의미

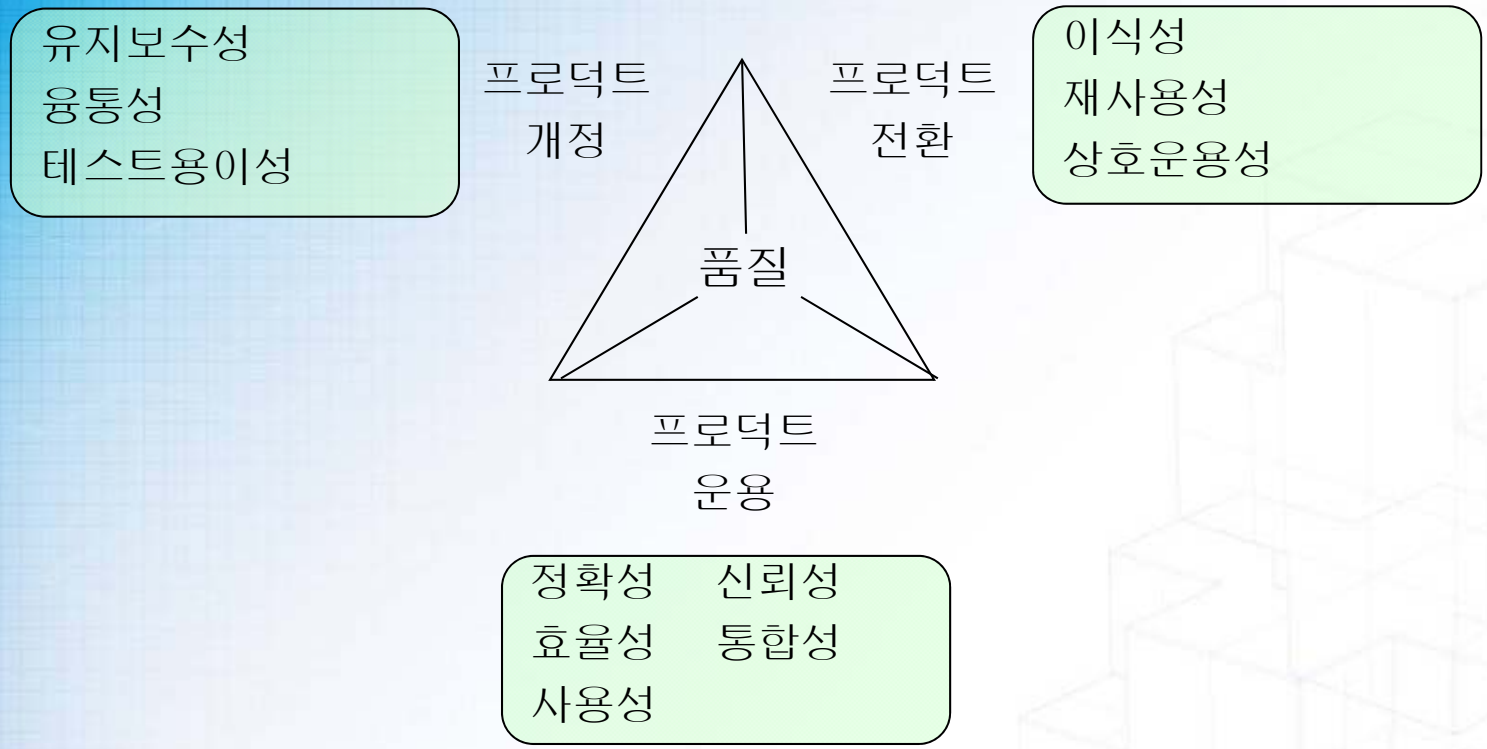
❑ 두 가지 개념의 소프트웨어 품질



- 넓은 의미: 고객 만족(Denning)
  - 정확하고 성능이 좋은 것은 물론이며 고객이 원하는 바로 그것, 어떤 탁월성을 보여서 사용하면서 만족을 얻을만한 것이 되어야 한다.
    =>토탈 상품, 시그마 운동
- 좁은 의미:
  - 개발자의 입장으로 국한 할 수 있고 개발자가 생각하는 품질의 개념은 소프트웨어가 요구를 만족하느냐 못하느냐의 여부

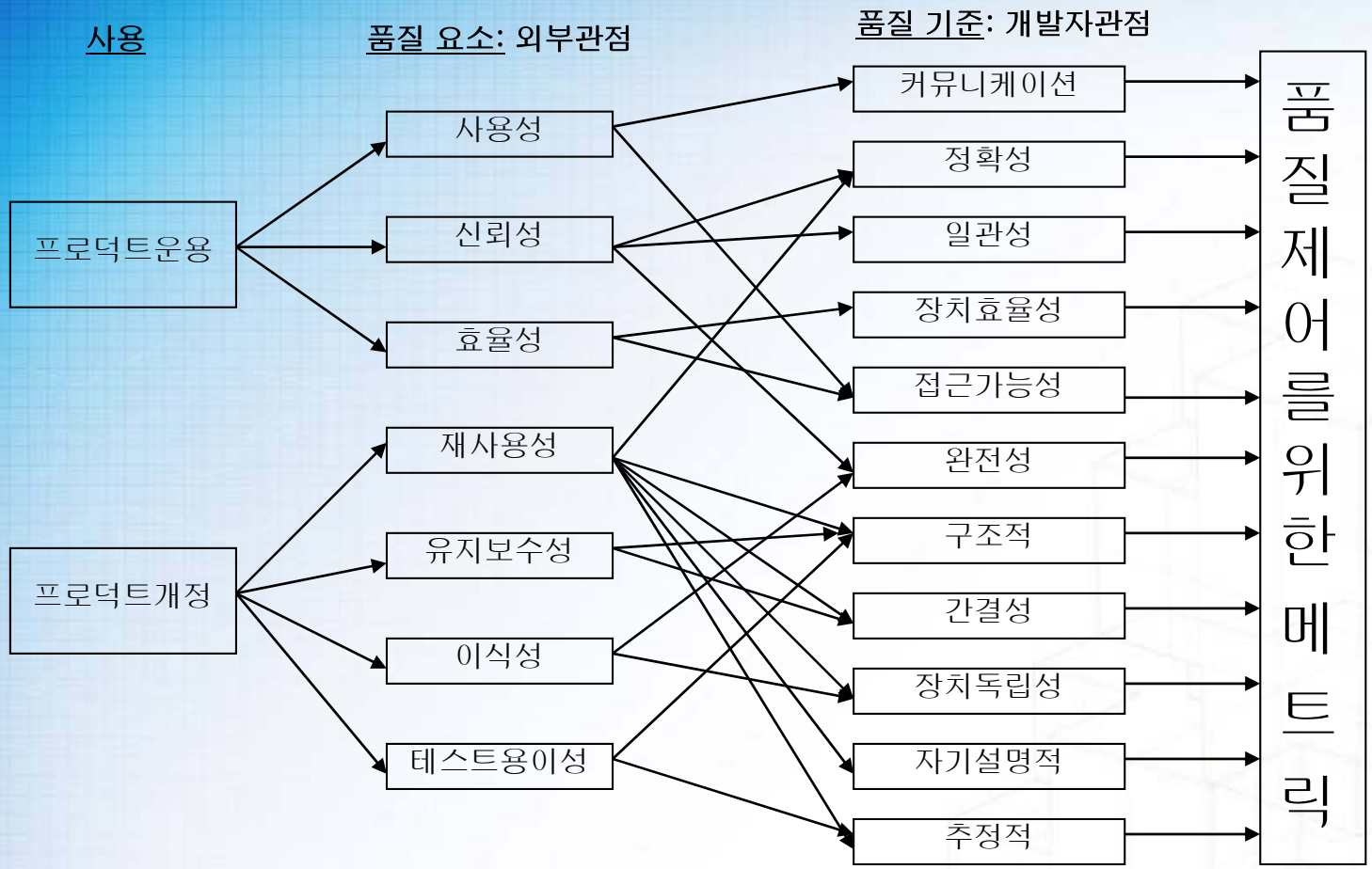
품질 요소(1/3)

❑ 소프트웨어 품질 요소



품질 요소(2/3)

❑ 소프트웨어 품질 특성의 세가지 차원

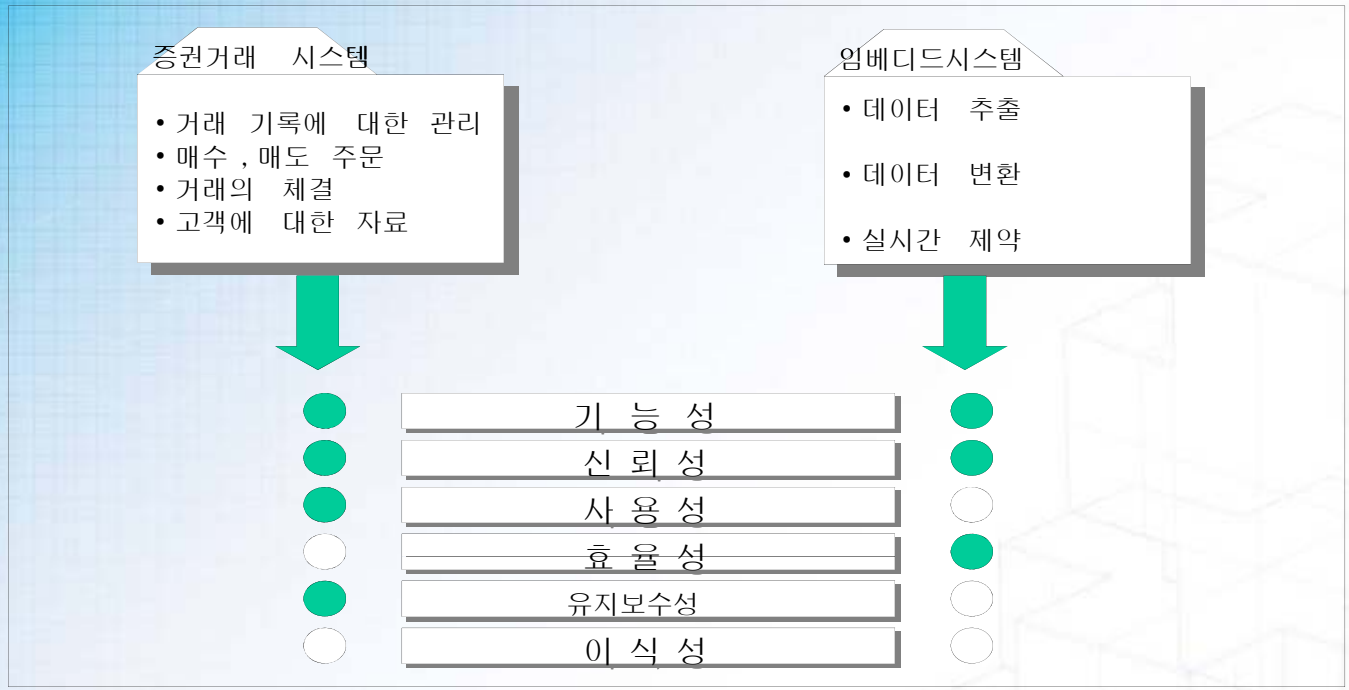


품질 요소(3/3)

- ISO 9126의 소프트웨어 품질 특성

소프트웨어 특성과 품질(1/2)

- 소프트웨어 특성에 따라 품질에 대한 요구는 매우 다를 수 있다.
  - Ex1) 증권 거래 시스템 → 기능성, 신뢰성, 사용성, 변경의 용이성
  - Ex2) 임베디드 시스템 → 신뢰성, 효율성



- 소프트웨어 유형에 따른 품질 특성 중요도의 차이

소프트웨어 특성과 품질(2/2)

❑ 소프트웨어 유형과 품질의 관계

- ❍ 소프트웨어의 결함이 인명과 재산에 치명적인 시스템
  - 신뢰성, 정확성, 테스트 용이성: 원자력발전소, 방사선 치료기 등
- ❍ 긴 수명을 요하는 시스템
  - 유지보수성, 이식성, 융통성
- ❍ 실시간 응용 분야의 소프트웨어
  - 효율성, 신뢰성, 정확성
- ❍ 보안이 중요한 응용 시스템
  - 다른 시스템과의 연동이 필요한 상호 운용성

프로세스 품질

❑ 품질 획득의 접근 방법

- ❍ 표준적인 개발 방법과 과정의 활용
  - 프로덕트의 품질을 보장 가능
- ❍ 가이드 및 평가 방법의 표준
  - ISO SPICE, CMM

품질 보증 활동

- 내용
  - 조직
  - 계획
  - 작업

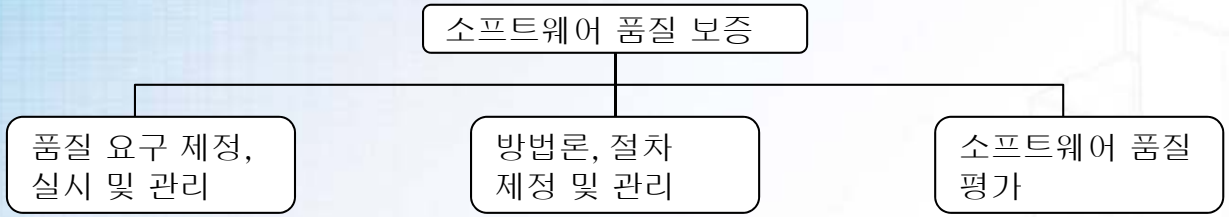
품질 보증 활동

❑ 개념

- ❍ 소프트웨어 제품이나 아이템이 정해진 요구에 적합하다는 것을 보장하는데 필요한 계획적이고 체계적인 활동[IEEE 610, 1991]

❑ 일반적인 품질 보증 작업

1. 소프트웨어 품질 확보를 위한 요구 제정
2. 소프트웨어를 개발, 운용, 유지보수하기 위한 방법론, 프로세스, 절차의 제정과 실행
3. 소프트웨어 제품이 품질을 평가하고 관련 문서, 프로세스, 품질에 영향을 미치는 액티비티를 평가하기 위한 방법론, 프로세스, 절차의 제정과 실행



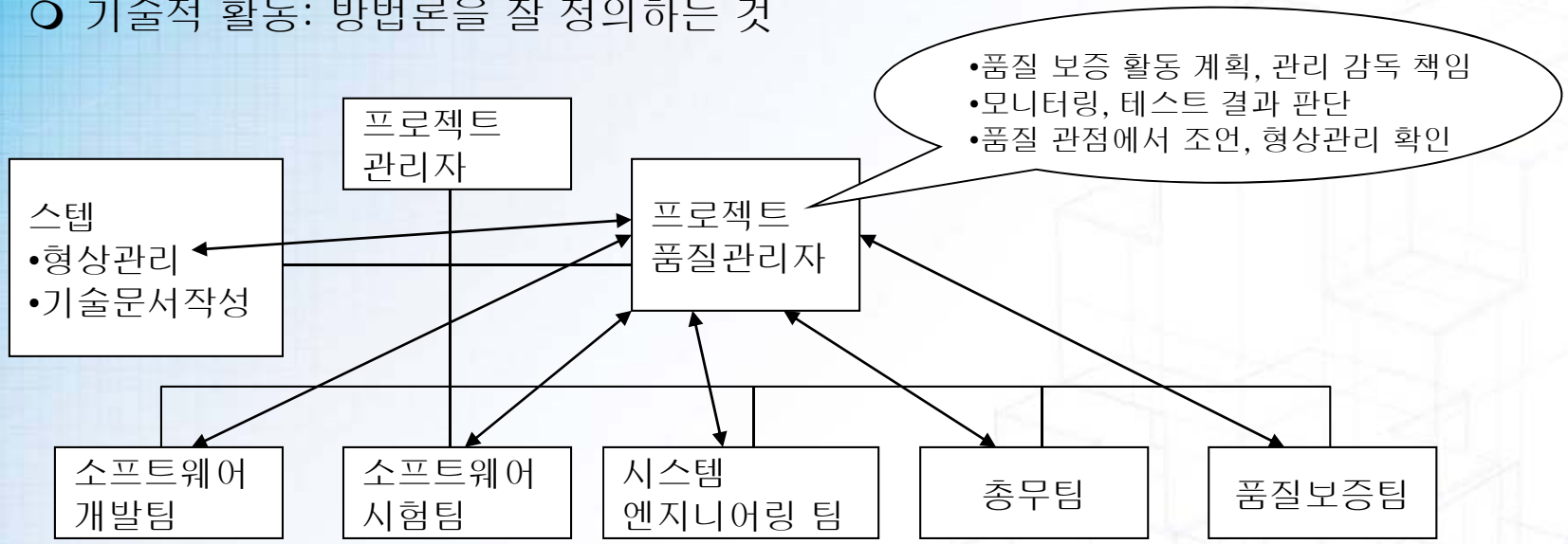
품질 요구 사항의 예
- 설계 요구사항
- 유지보수 요구사항
- 운용 요구사항

방법론
- 요구분석
- 설계
- 코딩
- 문서화
- 형상관리
- 운용 및 유지보수

품질 평가
- 프로덕트(개발 중, 개발 후)
- 활동(표준 준수여부)
- 방법론(기술적 정확성)

조 직

- 소프트웨어 품질 보증 활동과 개발 조직의 구분 필요
  - 품질 평가와 관리를 위한 객관적인 시각 요구
  - 기준 이하의 작업의 반복 또는 개선 유도
- 소프트웨어 품질 보증 작업의 활동
  - 관리적 활동: 개발조직의 표준화 방법론을 잘 따르도록 하는 것
  - 기술적 활동: 방법론을 잘 정의하는 것



품질 보증 조직

계 획

- 소프트웨어 품질 보증 활동의 시작

품질 보증 계획(IEEE 730)

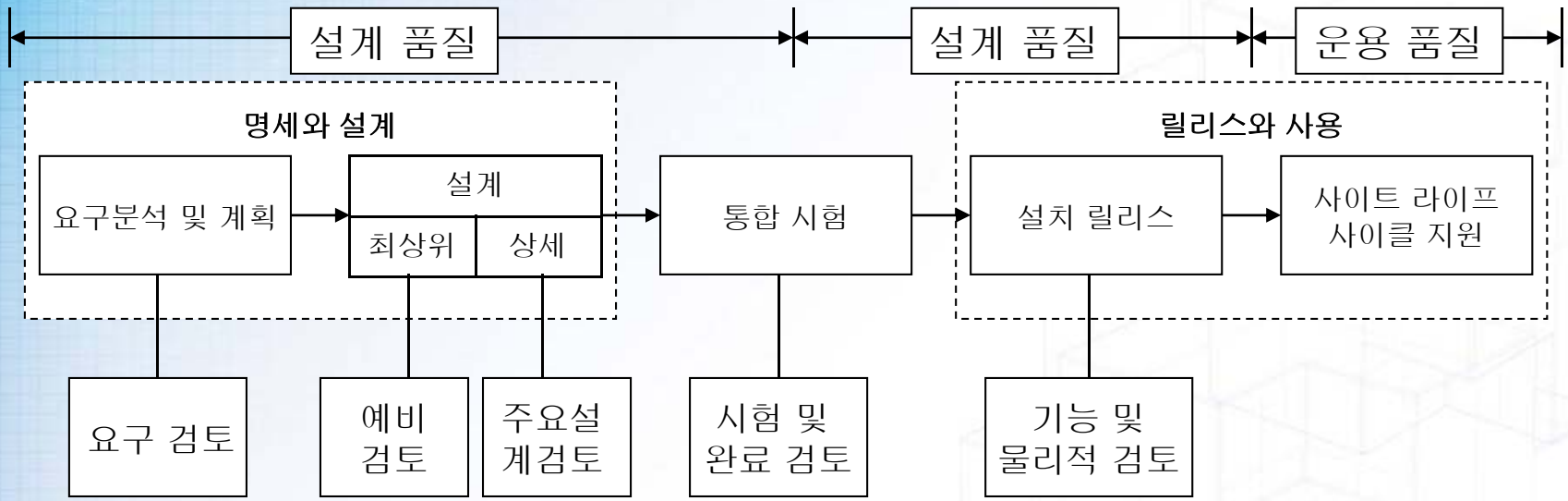
1. 계획의 목표
2. 관련문서
3. 관리
   - 3.1 조직
   - 3.2 작업
   - 3.3 책임
4. 문서화
   - 4.1 목적
   - 4.2 기본적으로 요구되는 문서
5. 표준, 실습, 관례, 메트릭
   - 5.1 목적
   - 5.2 내용
6. 검토와 검사
   - 6.1 목적
   - 6.2 요구되는 검토
7. 테스트
8. 문제 보고 및 수정 작업
9. 도구, 기술, 방법
10. 코드 관리
11. 미디어 관리
12. 공급자 관리
13. 기록 취합, 관리, 보관

작 업

❑ IEEE의 표준에서 제안하는 기본적인 검토작업(IEEE 730, 1989)

- ❍소프트웨어 요구 검토
- ❍예비 설계 검토
- ❍주요 설계 검토
- ❍소프트웨어 검사 및 검증 계획검토
- ❍기능 검토
- ❍물리적 검토
- ❍프로세스 검토
- ❍관리 검토
- ❍소프트웨어 형상관리 계획 검토
- ❍프로젝트 완료 후 검토

❑ 품질 보증을 위한 검토 작업



프로세스 품질

- 내용
  - 미국 정부 회계국 발표: 1991 전세계의 대형 프로젝트의 성공률이 1%
  - 프로세스: 개발을 위한 질서 있고 경험에 의한 규칙 있는 인력, 기술, 절차, 도구가 어우러져 통합된 작업
  - 프로세스에 의해서 소프트웨어의 품질을 높일 수 있다
  - "소프트웨어 시스템의 품질은 그것을 개발하는데 사용되는 프로세스의 품질에 좌우된다."(Humphrey)

- 소프트웨어 프로세스 평가 모델
  - CMM
  - SPICE

CMM

- 배경
  - 1992년 미국방성의 지원으로 설립된 CMU의 SEI가 제시
  - 1998년 버전 2.0으로 발전
- 목적
  - 현재의 프로세스 상태를 벤치마킹하고 어떤 부분을 향상시킬 것인지 전략을 선택하여 프로세스를 개선하려는 소프트웨어 개발 조직에 도움을 주기 위함
- 특징
  - 한정된 작업에 초점을 두어 공격적으로 활동함으로써 개발 조직의 계속적인 프로세스 향상 도모
  - 소프트웨어 엔지니어링 및 관리를 조직에 정착
- 사용 방법 두 가지
  - 미래의 고객이 소프트웨어 공급자의 강점과 약점 파악
  - 개발자 스스로 프로세스 능력 평가 후 개선 방향 설정

CMM 모델

레벨	초점	주요 프로세스 영역	결과
5 Optimizing (최적단계)	계속적인 개선	·프로세스 변경 관리 ·기술 변경 관리 ·결함 방지	생산성과 품질
4 Managed (관리단계)	프로덕트 및 프로세스 품질	·소프트웨어 품질 관리 ·정량적 프로세스 관리	
3 Defined (정의단계)	엔지니어링 프로세스	·조직 프로세스 집중 ·조직 프로세스 정의 ·동료 검토 ·교육 프로그램 ·그룹간 협력 ·소프트웨어 프로덕트 엔지니어링 ·통합 소프트웨어 관리	
2 Repeatable (반복단계)	프로젝트 관리	·소프트웨어 프로젝트 계획 ·소프트웨어 프로젝트 추적 및 감독 ·소프트웨어 하청 관리 ·소프트웨어 품질 보증 ·소프트웨어 형상 관리 ·요구 관리	
1 Initial (초보단계)	영웅적 개인		위 험

CMM 레벨 1 : Initial

- 소프트웨어 개발 과정이 임시 방편이며 무질서한 상태이다.
- 프로세스가 정확히 정의되어 있지 않고 개인의 능력에 의존한다.
- 소프트웨어 개발이나 유지보수를 위한 개발 조직의 환경 불안하다.
- 조직 내에 관리 부재로 소프트웨어의 품질 손상 우려가 있다.
- 심각한 경우는 프로젝트가 계획된 절차를 뛰어넘어 코딩과 테스트 작업으로 전환한다.
- 납기일을 맞추기 위하여 프로세스의 품질 보증 작업을 무시한다.
- 프로세스의 입력과 과정이 정해져 있지 않아 작업 결과의 예측이 불가능하다.
- 유사한 작업에 대한 프로젝트가 반복하여 진행되더라도 개발 생산성이나 품질에 큰 차이가 난다.

CMM 레벨 2 : Repeatable

- 기본적인 프로젝트 관리 프로세스를 확립한다.
- 비용, 일정, 기능에 대한 예측과 추적이 가능하다.
- 요구되는 프로세스 원칙들이 지켜져서 같은 문제에 대한 개발이 반복될 때 쉽게 성공이 가능하다.
- 조직은 소프트웨어 프로젝트 관리를 위한 정책을 입안하고 절차들을 실행한다.
- 새 프로젝트의 계획과 관리는 개인적인 경험보다는 과거의 데이터에 의존한다.
- 소프트웨어 프로젝트를 효과적으로 관리하는 프로세스가 전사적으로 잘 정착되어있다 .
- 프로젝트 관리자는 소프트웨어 비용과 일정, 기능을 추적할 수 있고 문제가 발생하면 바로 인식한다.
- 소프트웨어 요구와 이를 만족하기 위하여 개발된 결과물은 반드시 체크하고 일치하는지 확인한다.
- 소프트웨어 프로젝트 표준이 잘 정의되어 있고 조직이 표준을 잘 준수하고 있는지 확인한다.
- 원리 원칙을 준수한다.

CMM 레벨 3 : Defined

- 관리와 엔지니어링을 위한 소프트웨어 프로세스가 문서화되어 규격으로 만들어져 있다.
- 기관 전체가 표준 소프트웨어 프로세스를 따른다.
- 기관 내에 소프트웨어 엔지니어링 프로세스 그룹(Software Engineering Process Group)이 조직되어 소프트웨어 프로세스를 전담한다.
- SEPG는 스탭과 관리자를 위하여 부여된 역할을 잘 수행하기 위한 지식과 기술을 가르치는 교육도 담당한다.
- 표준화와 일관성이 잘 지켜진다.
- 소프트웨어 엔지니어링과 관리 활동이 안정적이며 성공적으로 되풀이 될 수 있다.
- 개발 공정, 비용, 일정, 기능이 통제되고 있고 소프트웨어 품질에 대하여 추적이 가능하다.
- 이러한 프로세스 능력은 소프트웨어 프로세스가 잘 정의되고 전체 조직이 작업과 역할, 의무를 잘 이해하고 있기 때문에 가능하다.

CMM 레벨 4 : Managed

- 소프트웨어 프로세스와 프로덕트의 품질에 대한 자세한 측정이 이루어진다.
- 즉 소프트웨어 프로세스와 프로덕트가 정량적으로 이해되고 통제된다.
- 조직은 소프트웨어 프로덕트와 프로세스에 대한 정량적인 목표를 세운다.
- 프로젝트에서의 모든 작업은 생산성과 품질 측면에서 측정하고. 프로젝트에서 정의된 프로세스에 대하여 데이터를 모으고 분석하여 소프트웨어 프로세스 데이터베이스에 저장한다.
- 잘 정의되고 일관된 측정 방법에 의하여 소프트웨어 프로세스를 측정한다.
- 프로세스 수행 성과의 편차를 허용하는 범위 안으로 줄여 프로덕트와 프로세스를 관리한다.
- 프로세스가 측정되고 정해진 범위 내에서 이루어지므로 프로세스 성과를 정확히 예측할 수 있다 .
- 기관은 이를 통하여 프로세스와 프로덕트에 대한 품질의 추세를 예측한다.
- 만일 정해진 범위를 벗어날 때는 바로잡기 위한 조치를 취하여 품질 예측 가능성이 높아진다.

CMM 레벨 5 : Optimizing

- 프로세스에 대한 정량적인 피드백과 획기적인 아이디어와 기술로 지속적으로 프로세스를 향상시킨다.
- 프로젝트 프로세스의 생산성 향상을 위하여 오래된 기술과 방법론은 적절히 퇴출시킨다.
- 전체조직이 지속적인 프로세스 개선을 위하여 초점을 맞춘다.
- 결함을 방지할 목적으로 프로세스의 약점과 강점을 미리 파악하는 수단을 가지고 있다.
- 소프트웨어 프로세스의 효율성에 대한 데이터를 사용하여 새로운 기술의 도입과 소프트웨어 프로세스의 변경에 의한 비용과 수익을 분석한다.
- 최고의 소프트웨어 엔지니어링 기술을 찾아내고 이를 이용하여 혁신을 추구한다.
- 소프트웨어 프로젝트 팀은 결함을 파악하여 그 원인을 분석한다.
- 소프트웨어 프로세스를 평가하며 결함이 재발되는 것을 방지하며 노하우를 다른 프로젝트에 전수한다.
- 개발 기관이 프로세스 능력의 범위를 향상시키기 위하여 계속 노력하며 이를 통하여 프로젝트 단위의 프로세스 성과도 향상된다.

CMM 각 영역의 공통 특징

❑ 각 프로세스 영역의 다섯 가지 공통 특징

❍ 실행 합의(commitment to perform)

- 해당 프로세스 영역에서 조직의 모든 사람 특히 경영관리자가 동의하고 법제화하여 그 업무를 실행할 수 있는 기본적인 정의와 합의

❍ 실행 능력(ability to perform)

- 프로젝트 단위 또는 기관 전체 조직이 프로세스를 완전하기 실행하기 위한 기본 조건
- 자원, 조직, 교육 등이 포함되어 있다

❍ 수행 활동(activities performed)

- 주요 프로세스 영역의 작업을 수행하는데 필요한 역할, 절차, 작업 수행, 추적, 피드백 등

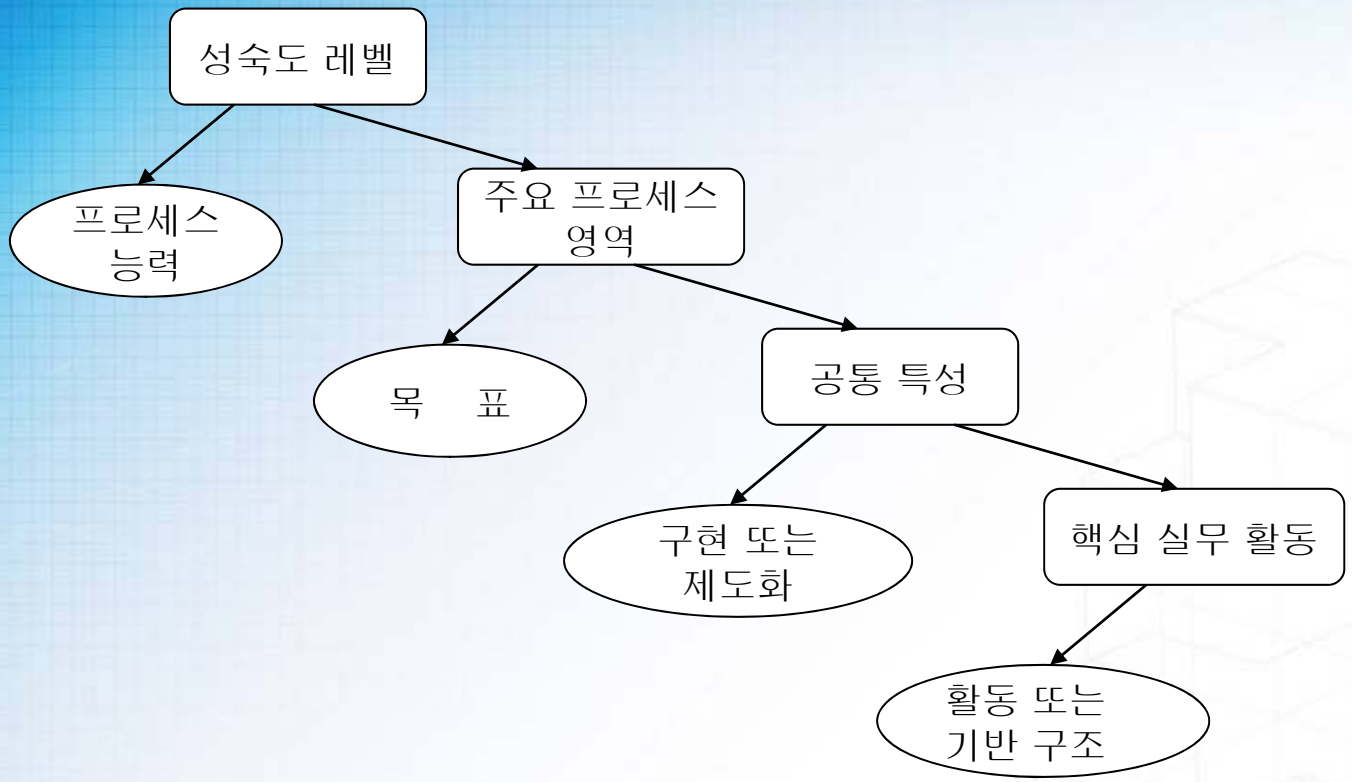
❍ 측정 및 분석(measurement and analysis)

- 프로세스 측정과 분석의 필요성 기술
- 실행된 활동에 대하여 상태와 효율성을 검토할 수 있는 측정 방법의 일부 사례 기술

❍ 구현 검증(verifying implementation)

- 수행한 작업이 정의된 표준 프로세스와 부합된다는 것을 보증하기 위한 작업을 기술
- 관리자와 품질보증 조직이 검토하고 감사하는 작업의 포함

CMM 구조



CMM 핵심 실무 활동(Key Practices)

- 필요한 업무 활동을 자세히 표현
- 가장 기본적이고 중요한 기반구조 및 활동을 나타냄
- 예) 프로젝트 계획 영역의 핵심 실무 활동
  - 비용산정, 라이프사이클의 정의, 일정과 투입 자원에 대한 계획의 문서화
- 핵심 실무활동에 대하여 CMM은 활동 항목만 나열하고 있으며 이를 어떤 방법으로 수행하여야 하는지는 언급하지 않고 있다 [Paulk, 1994].

SPICE

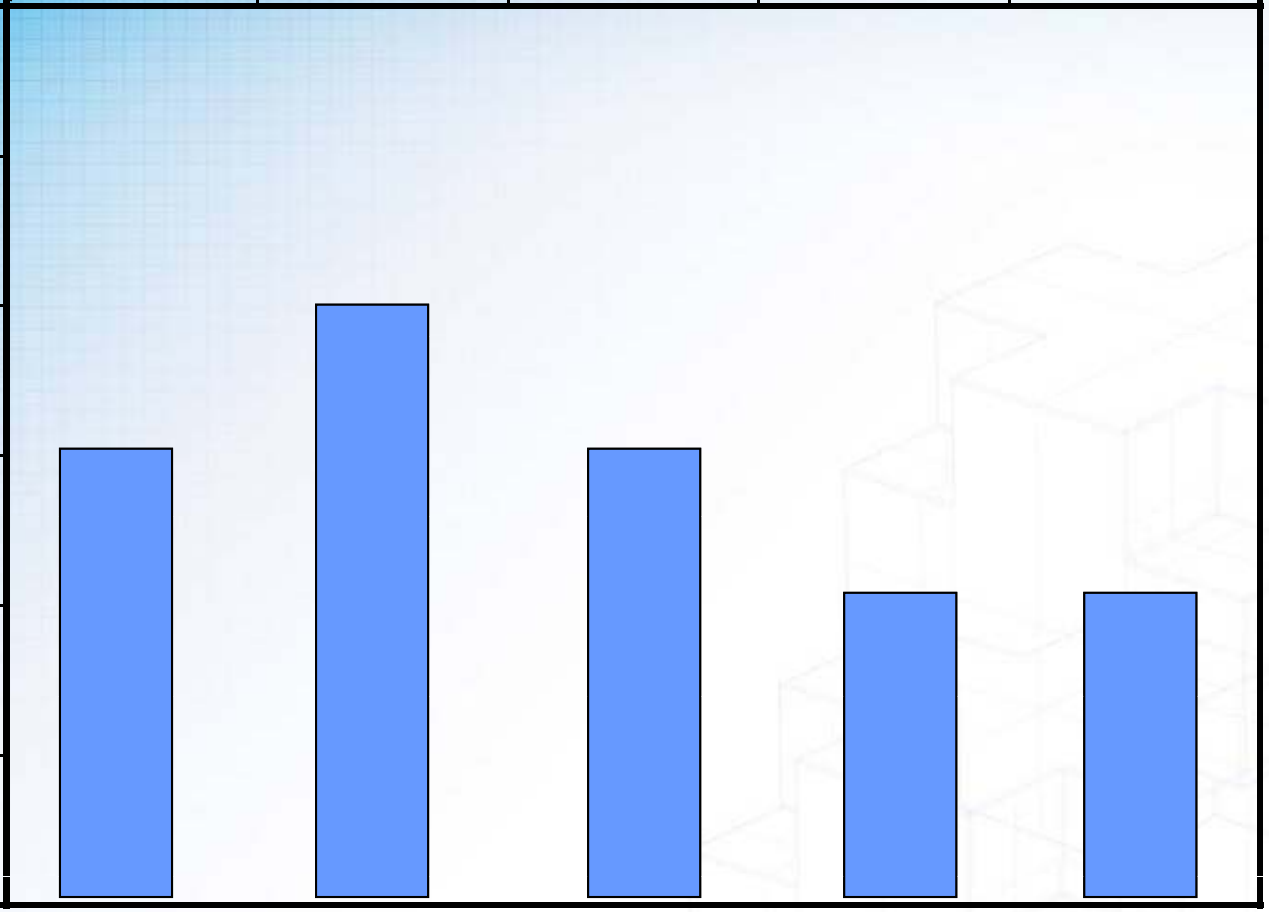
- Software Process Improvement and Capability dEtermination
- 소프트웨어 프로세스 평가를 위한 국제 표준을 제정하는 국제적인 표준화 프로젝트
- 배경
  - 1995년 ISO/IEC 15504라는 규격으로 완성
- 목적
  - 개발 기관이 프로세스 개선을 위하여 스스로 평가
  - 기관에서 정한 요구조건을 만족하는지 개발 조직이 스스로 평가
  - 계약을 맺기 위하여 수탁 기관의 프로세스를 평가

SPICE vs CMM

- 유사점
  - 프로세스 심사를 위한 참조모형을 제공
    - 개발 성숙도에 따라 차별화된 수준을 정의
    - 각 수준의 특징을 제시하여 기관의 수준을 판단 기준 제공
- 차이점
  - 성숙도 레벨과 심사 영역의 구분
    - CMM: 레벨 1부터 5까지 5개의 성숙도 주순을 정의
    - SPICE: 레벨 0부터 5까지의 6개의 수준으로 나누어 정의
  - 능력 평가 방법
    - CMM: 어떤 기관의 프로세스 능력을 여러 분야에 걸쳐 평가하여 하나의 레벨로 평가하는 일차원적인 구조
    - SPICE: 각 프로세스 영역마다 능력에 대한 평가를 별도로 할 수 있는 이차원적인 구조

SPICE의 이차원 모델

영역 / 레벨	고객-공급자 프로스세	엔지니어링 프로세스	지원 프로스세	관리 프로세스	조직 프로세스
5 Optimizing (최적 단계)					
4 Predictable (예측 단계)					
3 Established (확립 단계)					
2 Managed (관리 단계)					
1 Performed (실행 단계)					
0 Incomplete (미완성 단계)					



SPICE의 이차원 모델

- 고객 공급자 프로세스
  - 소프트웨어를 개발하여 고객에게 제공하고 소프트웨어를 정확하게 운용하고 사용하도록 지원하기 위한 프로세스
    - 예) 발주, 공급자 선정, 고객 인수, 요구사항 도출, 공급, 운영 등
- 엔지니어링 프로세스
  - 시스템과 소프트웨어 제품을 개발하는 모든 프로세스, 즉 요구분석, 설계 및 실험, 구축, 통합 등의 프로세스
    - 예) 요구분석, 설계 및 실험, 구축, 통합 등
- 지원 프로세스
  - 문서화, 형상관리, 품질보증, 검증, 확인, 검토 등 개발활동을 지원하는 프로세스
- 관리 프로세스
  - 일반적인 소프트웨어 프로젝트에서 일어나는 관리 활동
    - 예) 프로젝트 관리, 품질 관리, 위험 관리 등
- 조직 프로세스
  - 조직의 업무 목적을 수립하고 조직이 업무 목적을 달성하기 위하여 도움을 주는 프로세스
    - 예) 프로세스의 정의, 심사, 개선, 인적자원 관리, 기반구조, 측정, 재사용

SPICE 6단계 능력 수준(1/2)

- 레벨 0 (미완성 단계)
  - 목표 달성에 실패하는 경우가 많다
  - 쉽게 생각할 수 있는 프로세스의 작업산출물이나 결과가 존재하지 않는 수준이다
- 레벨 1 (실행 단계)
  - 프로세스가 성공적으로 완수하기 위하여 철저하게 계획하거나 추적되지 않을 수도 있다
  - 철저한 관리가 수행되어야 한다는 것을 인지하고 있으며 더 강화된 노력에 대하여 동의한다
  - 구별된 작업산출물이 존재하고 이들을 근거로 목표 달성 여부가 결정된다
- 레벨 2 (관리 단계)
  - 정해진 절차에 따라 이루어져 산출물을 내며 모든 작업이 계획되고 추적된다
  - 산출물은 명시된 표준과 요구사항에 부합한다
  - 레벨 1과의 차이점
    - 정해진 시간과 자원 한도 안에서 프로세스를 수행, 정해진 품질 요구사항을 만족하는 산출물
- 레벨 3 (확립 단계)
  - 소프트웨어 엔지니어링 원리에 근거하여 프로세스를 정의, 이를 이용하여 프로세스를 수행하고 관리한다
  - 정의된 프로세스가 표준화되어 있고 문서화되어 있다
  - 프로세스를 수행할 때는 표준 프로세스를 알맞게 조정하여 승인 받은 후 사용한다
  - 레벨 2와의 차이점
    - 프로세스에 정의된 성과를 달성할 수 있도록 표준으로 정의된 프로세스를 사용한다

SPICE 6단계 능력 수준(2/2)

- 레벨 4 (예측 단계)
  - 일정한 통제 범위 내에서 일관성 있게 수행한다
  - 프로세스를 수행한 후 자세한 측정값을 수집하고 분석한다
  - 프로세스 능력의 정량적 이해, 수행 예측 그리고 관리할 수 있다
  - 작업 후 산출물의 품질도 정량적으로 알 수 있다
  - 레벨 3과의 차이점
    - 정해진 성과를 달성하기 위하여 표준 프로세스를 수행하며 그 결과는 일정한 한도 내에서 통제된다

- 레벨 5 (최적화 단계)
  - 현재 프로젝트만이 아니라 미래에 수행될 프로세스에 대하여도 목표를 잘 만족시킬 수 있다
  - 수립된 목표에 따라 프로세스를 수행할 때 달성할 정량적인 효율 목표를 설정한다
  - 목표에 대하여 지속적으로 프로세스에 대한 모니터링이 가능하다
  - 결과를 분석하여 지속적인 개선을 할 수 있다
  - 최적화를 위한 혁신적인 아이디어와 기술은 시범적으로 적용하여 도입
  - 레벨 4와의 차이점
    - 현재와 미래의 프로젝트를 성공적으로 완성시키기 위하여 정의된 표준 프로세스가 계속적으로 향상되기 위하여 역동적인 변화 및 시도가 있다

프로덕트 품질 측정

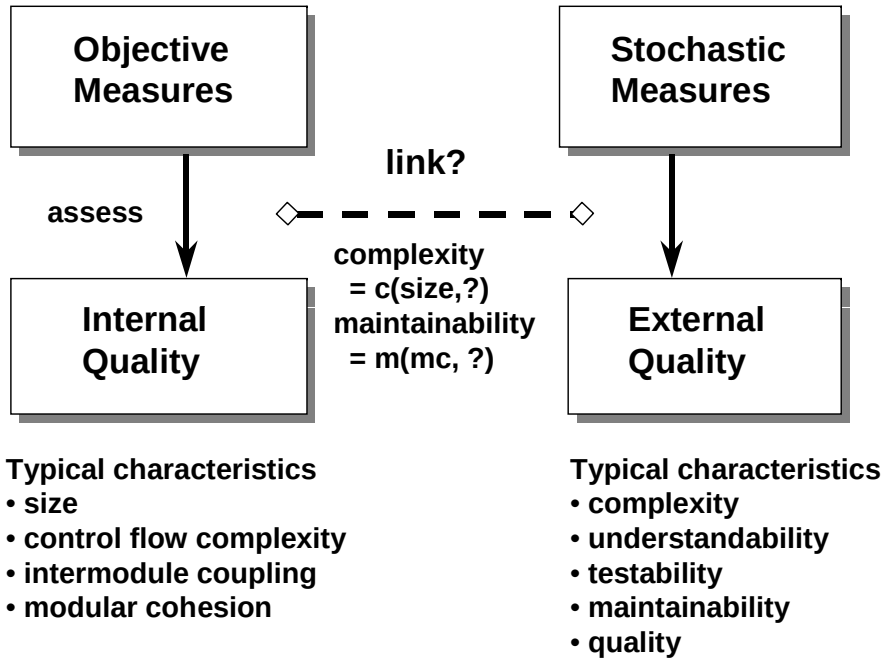
- 메트릭과 수치적 가시화 방법이 필요

- 내부 품질
  - 외부 환경의 영향을 받지 않고 객관적으로 측정할 수 있는 특성
  - 프로그램의 크기, 제어 흐름의 복잡도, 모듈의 응집력, 결합도 등 객관적으로 측정 가능

- 외부 품질
  - 외부 환경 요소에 따라 달라질 수 있다
  - 정확한 메트릭이 어렵다
  - 외부 환경에 영향을 받는다

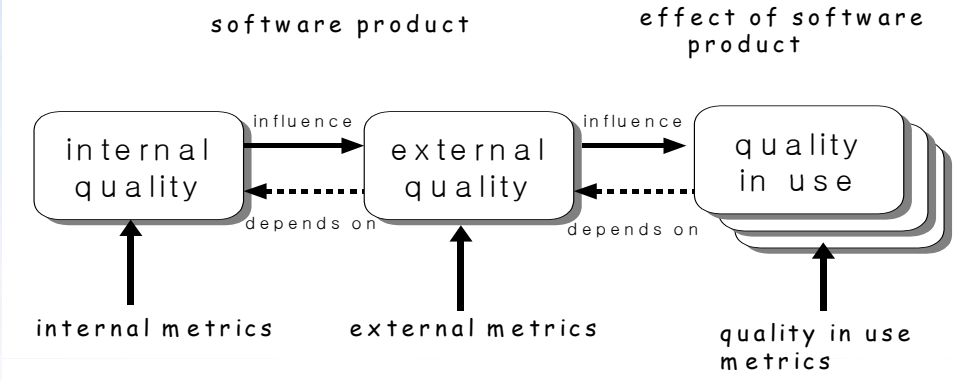
프로덕트 품질 측정

- 내부와 외부 품질 요소
  - 예측 모델(predictive quality)
    - 내부 요소와 외부 품질과의 함수관계



프로덕트 품질 측정

- 품질 특성의 정의
- ISO 9126 시리즈
  - 내부 메트릭
    - 개발 단계에서 생성되는 요구분석, 설계, 원시코드와 같은 비실행 결과물
  - 외부 메트릭
    - 완성된 소프트웨어의 성능, 오류 발생, 사용용이성 등 시스템의 동작을 측정
  - 실용 환경에서의 품질(quality in use)
    - 특정 환경에서 사용자가 원하는 수준이 품질 특성
- 품질 수준



프로덕트 품질 측정

- 메트릭 스케일
  - 메트릭 개발을 위해 스케일 타입과 측정 타입을 정한다
  - 메트릭 스케일

스케일 타입	매핑	예
Nominal(일대일)	M' =F(M)	분류, 레이블링
Ordinal(등급)	M' =F(M) M(x)>=M(y) 이면 M' (x)>=M' (y)	IQ, 우선순위, 난이도, 청정도
Interval(간격)	M' =aM+b(a>0)	온도, 시간
Ratio(비율)	M' =aM<(a>0)	길이, 무게, 시각, 크기
Absolute(절대)	M' =M	카운트

  - 소프트웨어에 적용되는 일반적 메트릭
    1. 크기를 측정하는 타입: 소프트웨어의 추상적 기능단위, 원시코드 한 줄, 모듈 단위
    2. 시간을 측정하는 타입: 시스템 성능 측정, 개발 노 측정
    3. 카운트 타입: 오류의 수, 프로그램의 구조적 복잡도, 변경 횟수 등

프로덕트 품질 측정

❑ ISO의 외부 품질 메트릭 예

품질특성	부특성	메트릭	측정방법	범위	스케일
기능성	적합성	기능 타당성	X=1-문제유발기능의 수/전체기능의 수	0<=X<=1	절대
		기능 구현 완전성	X=1-빠진 기능의 수/전체기능의 수	0<=X<=1	절대
		기능 구현 범위	X=-1잘못 구현된 기능 수/전체 기능 수	0<=X<=1	절대
	정확성	예측 정확성	X=1-예측한 결과가 차이 나는 회수/운영 시간	0<=X	비율
		계산 정확성	X=부정확한 계산 회수/운영 시간	0<=X	비율
		정밀성	X=예상과는 다른 자릿수/운영 시간	0<=X	비율
	상호 운용성	데이터 교환성 (자료형태)	X=성공적으로 데이터가 교환된 회수/데이터가 교환된 총 수	0<=X<=1	절대
		데이터 교환성 (사용자 접근 시도)	X=1-교환에 실패한 회수/교환 시도 총 회수	0<=X<=1	절대
	보안성	접근 감시	X=시스템 접근 내역 기록 회수/시스템 접근 회수	0<=X<=1	절대
		접근 통제	X=불법적인 접근 적발 회수/불법 오퍼레이션 수	0<=X<=1	절대
	준수성	기능 준수성	X=1-구현되지 않은 기능 수/기능 준수 아이템 수	0<=X<=1	절대
		인터페이스 표준 준수성	X=적합하게 구현된 인터페이스 수/적합성이 요구되는 인터페이스 수	0<=X<=1	절대
신뢰성	성숙성	예측된 결함 빈도	X=ABS예측된 오류-발견된 오류)/제품 크기	0<=X	절대
		결함 발생율	X=발견된 결함의 수/수행된 테스트 수	0<=X	절대

인스펙션

❑ 배경

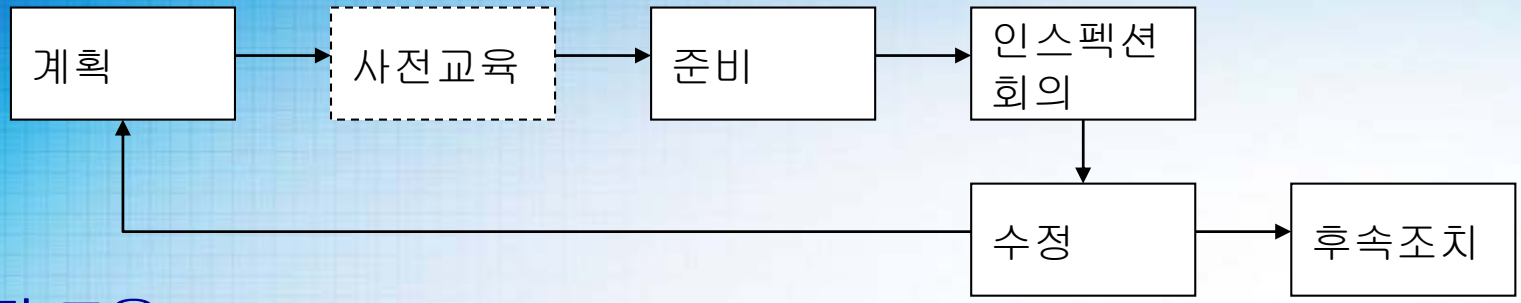
- 1976년 Fagan의 논문 발표
- 품질 개선과 비용 절감을 위한 기법으로 사용
- 소프트웨어 품질 보증에 탁월한 효과
- 사전에 적은 노력으로 결함 발견

인스펙션의 목적

- 목적
  - 준비하는 동안 예상되는 결함을 찾아내고 이를 확인
  - 발견된 아이템이 실제 결함이라는 사실을 확인
  - 결함이 있다는 사실을 기록
  - 개발자가 수정하도록 기록을 제시

- 부수적인 목적
  - 요구에서 설계로 추적하기 위하여
  - 다음 개발 단계를 위한 기술적 기초를 제공하기 위하여
  - 프로그래밍 품질을 향상시키기 위하여
  - 라이프 사이클 비용을 낮추기 위하여
  - 테스트 작업의 효율을 높이기 위하여
  - 프로그램의 유지보수성을 인식하기 위하여
  - 소프트웨어 관리의 시작과 종료에 도움을 주기 위하여

인스펙션의 과정(1/3)



1. 사전 교육

- 인스펙션 팀을 정한다.
- 팀 구성원이 인스펙션을 잘 준비할 수 있는지 확인한다.
- 인스펙션을 할 대상이 지켜야 할 규격을 준수하고 있는지 확인한다.
- 체크 리스트를 이용하여 인스펙션 착수조건에 부합하는지 결정한다.
- 사전교육의 필요성을 결정한다.
- 인스펙션 회의 장소를 결정하고 확보한다.
- 인스펙션 회의 일정과 장소를 예약한다.
- 인스펙션 팀 구성원과 관계자에게 회의 시간과 장소를 알린다.
- 인스펙션 팀 구성원에게 검토 역할을 할당한다.

인스펙션의 과정(2/3)

2. 사전 교육
   - 인스펙션 팀 구성원을 대상으로 하는 간단한 교육
   - 소프트웨어를 간단히 설명
   - 어떤 작업이 어떻게 수행되었는지 설명
   - 어떤 인터페이스가 있으며 기능은 무엇인지 설명
   - 사전 교육 실행 여부는 인스펙션 주재자의 판단

3. 준비
   - 인스펙션 자료를 받았고 인스펙션 회의를 통고 받았을 때 부터 시작
   - 모든 자료는 회의 5일 전에 한번에 전달
   - 인스펙션 회의는 두 시간을 초과하지 말아야 한다
   - 인스펙션 팀 구성원은 발표할 자료 준비
   - 이해하기 어려운 부분을 표시하고 자료를 검토하여 결함 발견
   - 기록 양식에 기록
   - 준비에 소요되는 시간을 기록

인스펙션의 과정(3/3)

- 인스펙션 회의
  - 발견한 잘못된 점을 팀 구성원이 모여 논의
  - 인스펙션을 준비한 시간을 기록하고 발견한 것에 대해 토론
  - 결함으로 판단되면 기록하고 종류를 구별
  - 해결책은 논의하지 않는다(해결책은 프로덕트 작성자의 의무)
  - 회의 후에 참여자들의 결함 기록 회수
  - 발겸한 결함에 대해 종합하고 준비비간과 인스펙션 시간을 기록한 후 인트펙션의 재시행 여부, 인스펙션의 유형등을 작성자에게 알리고 폐회
- 수정
  - 작업을 면밀히 살펴보고 수정
  - 인스펙션이 다시 필요하다고 결정하면 인스펙션을 다시 준비
- 후속 조칙
  - 인스펙션에서 발견된 모든 결함이 수정되었는지 확인
  - 수정 결과는 직접 만나서 검사
  - 소프트웨어 품질 관리자게 종합 보고를 제출하고 종료

인스펙션의 종류(1/3)

- 시스템 설계 인스펙션
  - 목적: 모듈의 전반적인 설계나 기능을 점검
  - 주관심사: 소프트웨어 요구, 성능 명세 관련 사항과 인터페이스 설계
  - 시스템 설계 명세서에 포함된 인스펙션 대상
    - 설계 작업의 입력(계약 등)
    - 하드웨어 자원에 대한 기능 할당
    - 기능 흐름
    - 타이밍, 동기화에 대한 설계
    - 인테페이스
    - 분할 과정
    - 모듈 별 설계 정의

인스펙션의 종류(2/3)

- 상세 설계 인스펙션
  - 목적: 시스템 전체 설계의 목적을 반영하도록 각 모듈의 설계가 잘되었는가 검사
  - 설계 목표와 설계 작업의 관계

설계작업	설계 목표					
	신뢰성	정확성	테스트 가능성	유지보수성	변경성	확정성
하향적 구조	○		○			
모듈 크기			○	○		
낮은 복잡도	○		○	○	○	
구조적 프로그래밍	○		○	○	○	○
명령문의 그루핑			○	○	○	
심볼릭 매개변수			○	○	○	○
명명 규칙 준수			○	○	○	○
모듈 리스팅			○	○	○	○
여유 확보			○			○
성능 및 크기 분석	○		○			
정확도 모니터링		○	○			
소프트웨어 실행 경로 최소화	○		○	○	○	
설계 최적화		○				○
동적 모드 최소화	○		○	○	○	
설계 검토 강조	○	○	○			

인스펙션의 종류(3/3)

❑ 코드 인스펙션

❍ 전제

- 새로 작성된 프로그램이나 다른 시스템에서 개발되었으나 수정된 것
- 컴파일 오류가 없는 것이 확인된 프로그램만이 코드 인스펙션 가능

❍ 인스펙션의 목적

- 코드가 요구 명세와 설계 명세 및 인터페이스 명세에 적합한지 검사하기 위하여
- 설계가 정확히 프로그래밍 언어로 바뀌었는지 점검하기 위하여
- 코드의 품질을 동료가 점검하기 위하여
- 오류를 조기에 파악하기 위하여
- 코드가 모듈 사이의 인터페이스 요구를 만족하는지 검사하기 위하여
- 모듈 테스트 명세를 미리 검토하기 위하여
- 적당한 테스트 도구와 환경이 있는지 확인하기 위하여
- 모듈 테스트를 착수할 수 있는지 결정하기 위하여
- 소프트웨어 프로덕트가 계약이나 국제 표준 또는 규칙에 맞는지 점검하기 위하여