Chap 12 품질

목 차

12.1 소개

- 품질을 높이는 활동

1. 테스트
   - 제품주기에서 테스트가 너무 늦게 수행된다.
   - 테스트는 좁은 차원만 다룬다.
   - 테스트는 코드 품질만 향상시킨다.
2. 리뷰
   - 테스트를 보완, 개발 초기에 검토할 수 있어 오류가 조기에 발견
3. 품질보증
   - 개발자와 협력하여 소프트웨어 개발의 적절한 표준과 절차를 정의
   - 검토 및 감사를 통해 업무를 모니터링하여 확인
   - 품질 목표를 향한 진행 상황에 대해 상급 관리자 및 기타 이해 관계자에게 피드백을 제공

품질 관리

- 품질 계획, 품질 관리, 품질 보증, 검증 및 여러 가지 품질 관련 프로세스
- 품질 목표를 설정하고 목표 달성을 위한 프로젝트 실행을 관리 및 통제하기 위한 모든 활동
  - 측정
  - 프로세스 평가 및 개선

품질 개념

- 품질에 대한 다양한 관점
  - 관점에 따라 다른 정의
  - 품질에 대한 정의에 따라 품질 관리 활동이 달라짐
- 고객 만족
  - 사용자 만족도는 제품의 전반적 요소를 기반으로 하며 품질은 그중 하나

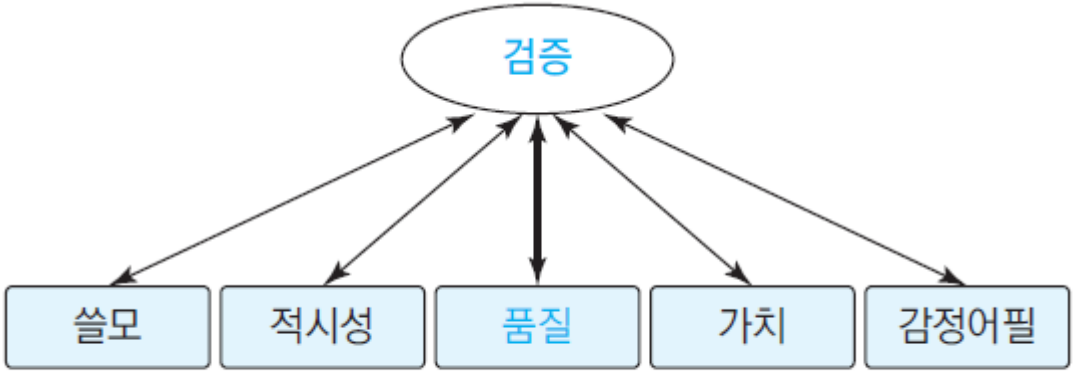


그림 12.3 품질과 고객만족

품질 관리

- 요구 적합성
  - 모호하지 않음
  - 지정된 요구 사항과 디자인을 준수하는 제품이 높은 품질의 제품
  - 일정 수준 이상의 고급이라는 의미가 없음
- 제품 품질
  - 여러 속성의 집합이며 이들이 품질을 결정
  - 디지털 카메라의 품질을 결정하는 세 가지 속성(해상도, 광 감도, 프레임 속도)

소프트웨어 품질

- 품질
  (1) 시스템, 구성 요소, 프로세스가 지정된 요구 사항을 충족시키는 정도.
  (2) 시스템, 구성 요소, 프로세스가 고객 또는 사용자 요구나 기대를 충족시키는 정도

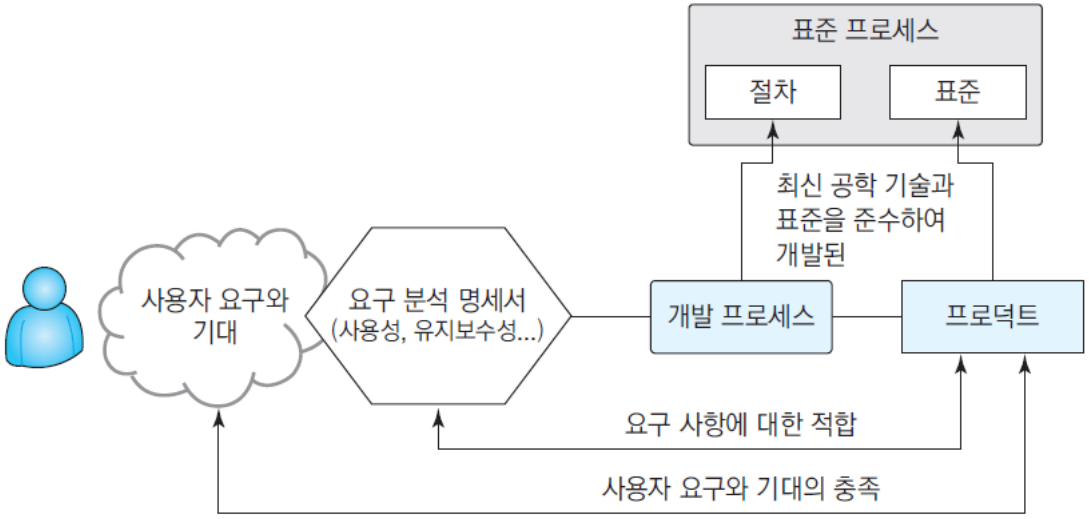


그림 12.4 소프트웨어 품질

12.2 품질 모델

- 소프트웨어에 대한 작업 관점이 어디 있느냐에 따라 품질 속성의 관심이 달라짐

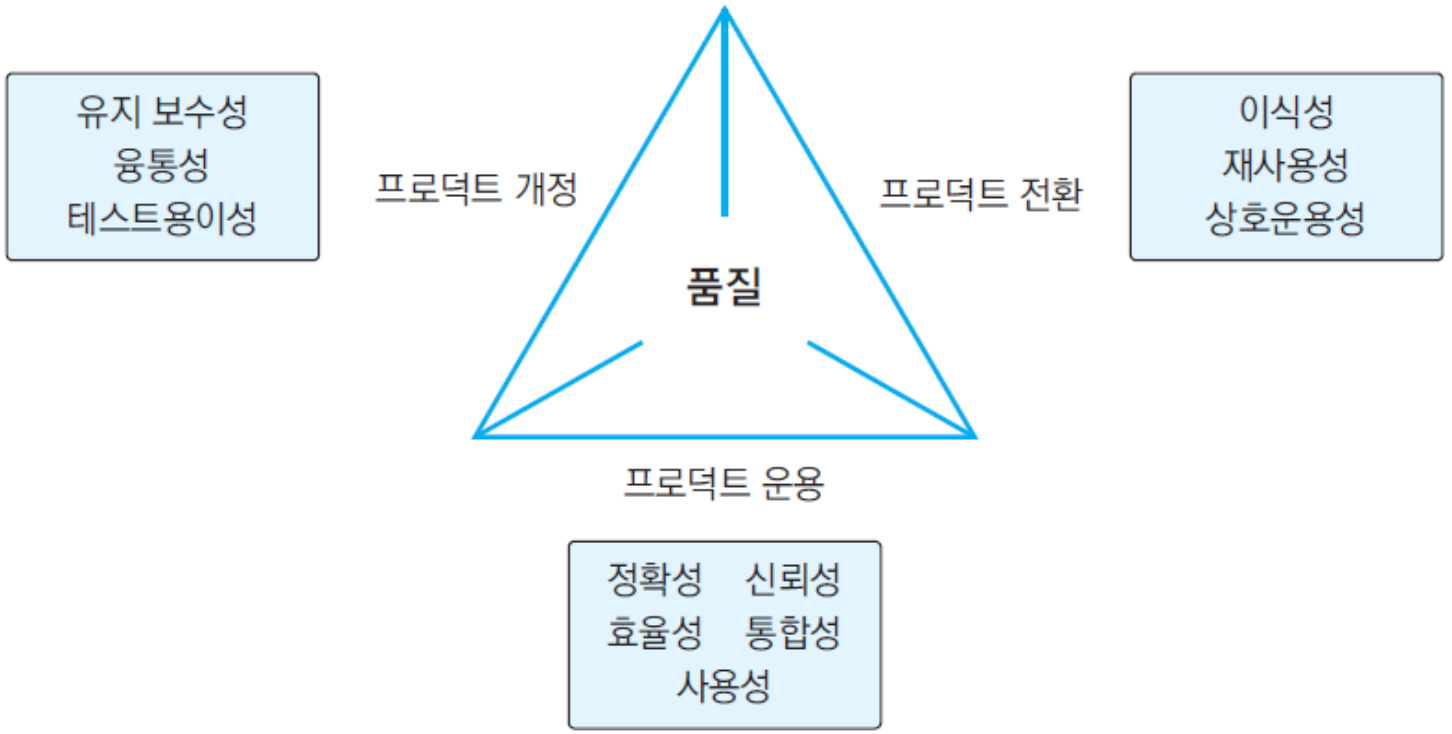


그림 12.5 소프트웨어 품질 속성

품질 특성

- 소프트웨어의 품질의 특성은 세가지 차원이 존재
  - 품질 요소(factor) : 사용자에 의한 외부 관점
  - 품질 기준(criteria) : 개발자 측면의 내부 관점
  - 메트릭 차원 : 품질을 제어

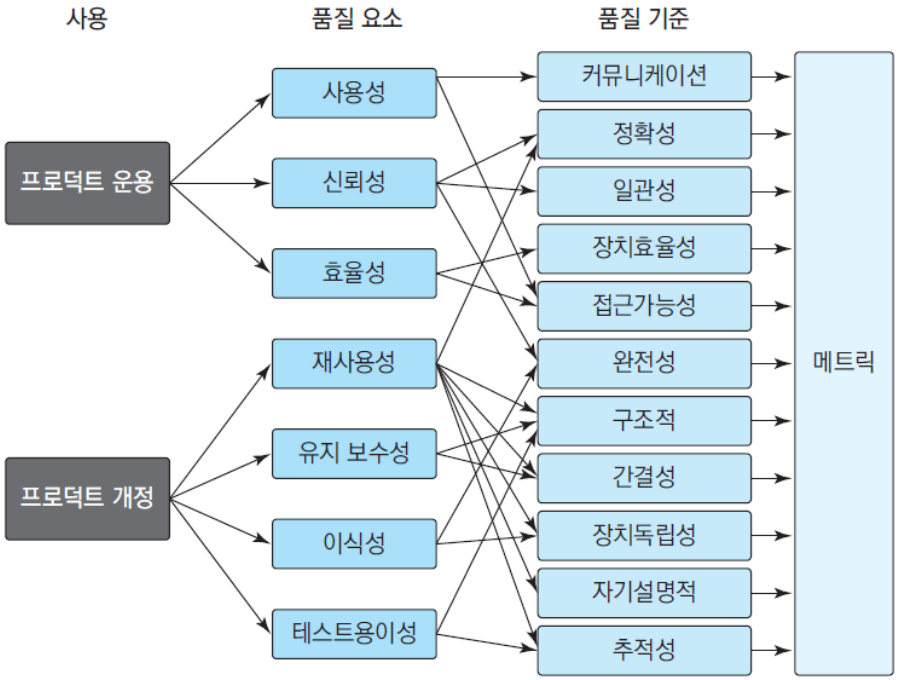


그림 12.6 소프트웨어 품질 특성의 세 가지 차원

품질 모델

- ISO/IEC 9126
  - 소프트웨어가 가질 수 잇는 여러 가지 품질 특성을 정의
  - ISO에서는 여섯 가지의 품질 특성을 정의
- ISO와 IEEE에서는 품질 속성의 계층을 다르게 정의

그림 12.7 ISO 9126과 IEEE 1061의 품질 모델

품질 속성

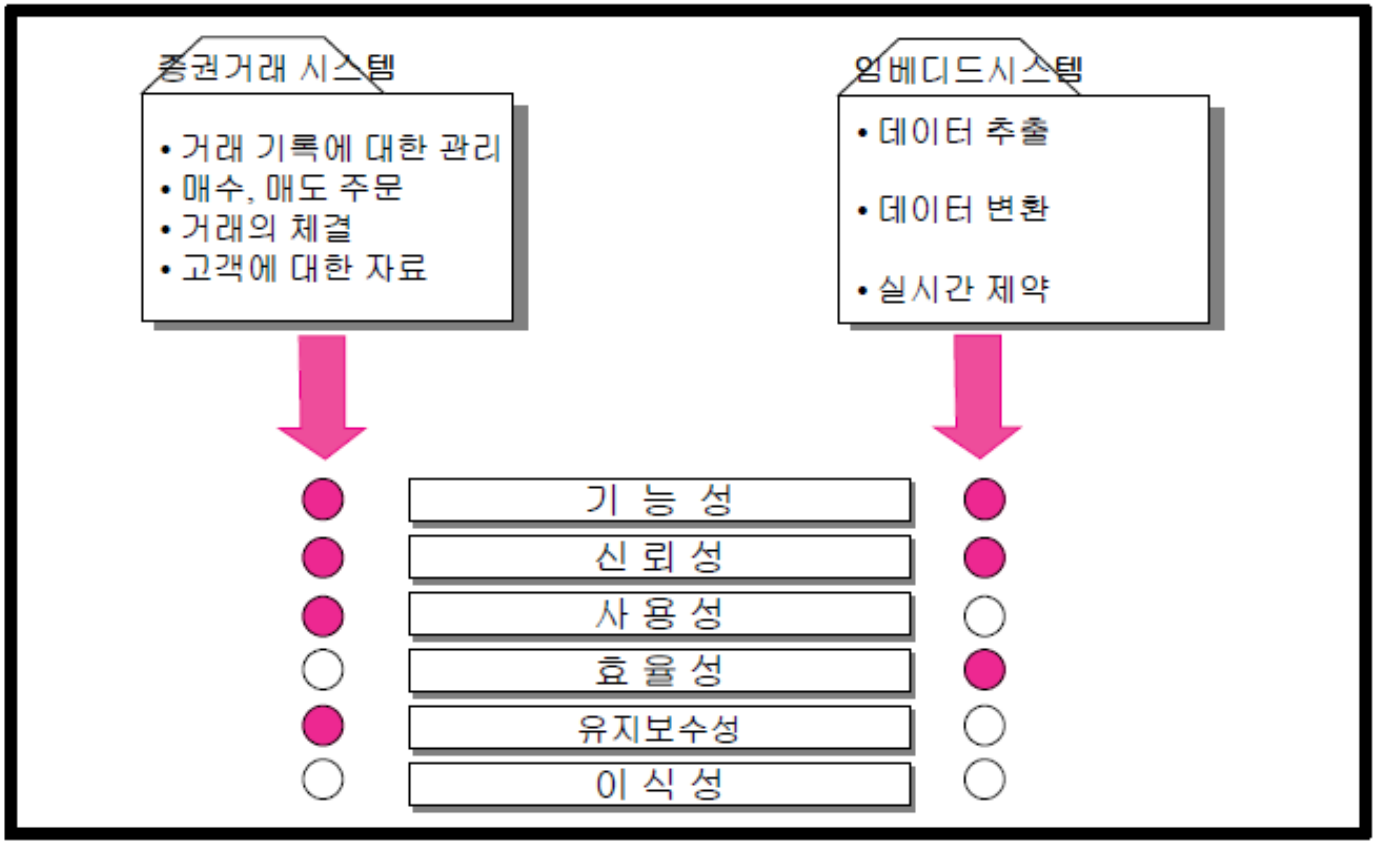
- 신뢰성(reliability) – 소프트웨어에 요구된 기능을 명시된 조건 하에 실행하여 정확하고 일관성 있는 결과를 생성하는 능력
- 강인성(robustness) – 소프트웨어에 요구된 기능을 어렵거나 예외적인 환경에서 수행할 능력
- 효율성(efficiency) – 소프트웨어에 요구된 기능을 최소의 시간과 자원을 사용하여 원하는 결과를 생성하는 능력
- 상호운용성(interoperability) – 소프트웨어가 다른 소프트웨어와 정보를 교환하는 능력
- 유지보수성(maintainability) – 소프트웨어가 수리 및 진화될 수 있는 성질

품질 속성

- 테스트가능성 – 소프트웨어에 요구된 또는 적용될 수 있는 모든 형식의 평가
  - 인스펙션, 동료검토, 화이트박스 테스팅, 블랙박스 테스팅 등
- 이식성(portability) – 소프트웨어가 여러 운영 환경 및 플랫폼에서 실행될 수 있도록 변형이 가능한 성질
- 재사용성(reusability) – 소프트웨어가 확장이나 커스텀화 없이 유사한 또는 다른 배경에서 사용될 수 있는 성질
- 모듈성(modularity) – 소프트웨어가 모듈 컴포넌트의 통합이나 조정을 쉽게 만드는 성질

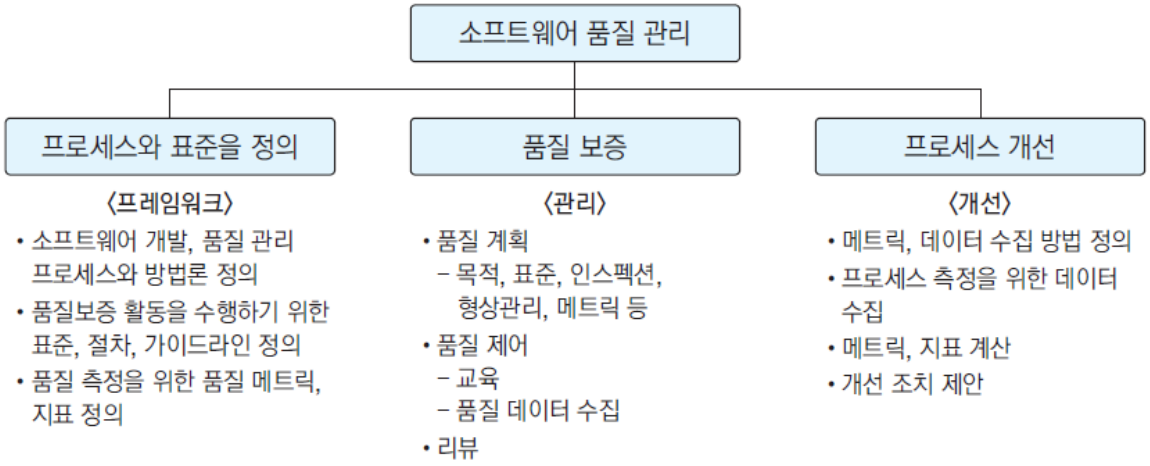
소프트웨어 유형과 품질

- 소프트웨어 유형에 따른 품질 특성 중요도의 차이



12.3 품질관리

- 소프트웨어 제품이나 아이템이 정해진 요구에 적합하다는 것을 보장하는 데 필요한 계획적이고 체계적인 활동
- 다양한 작업이며 미치는 영향이 크다
- 품질 관리 기능
  1. 프로세스와 표준의 정의
  2. 품질 보증
  3. 프로세스 개선



품질 보증 조직

- 관리적 활동
  - 개발 조직의 표준화 방법론을 잘 따르도록 하는 것
- 기술적 활동
  - 방법론을 잘 정의하는 것

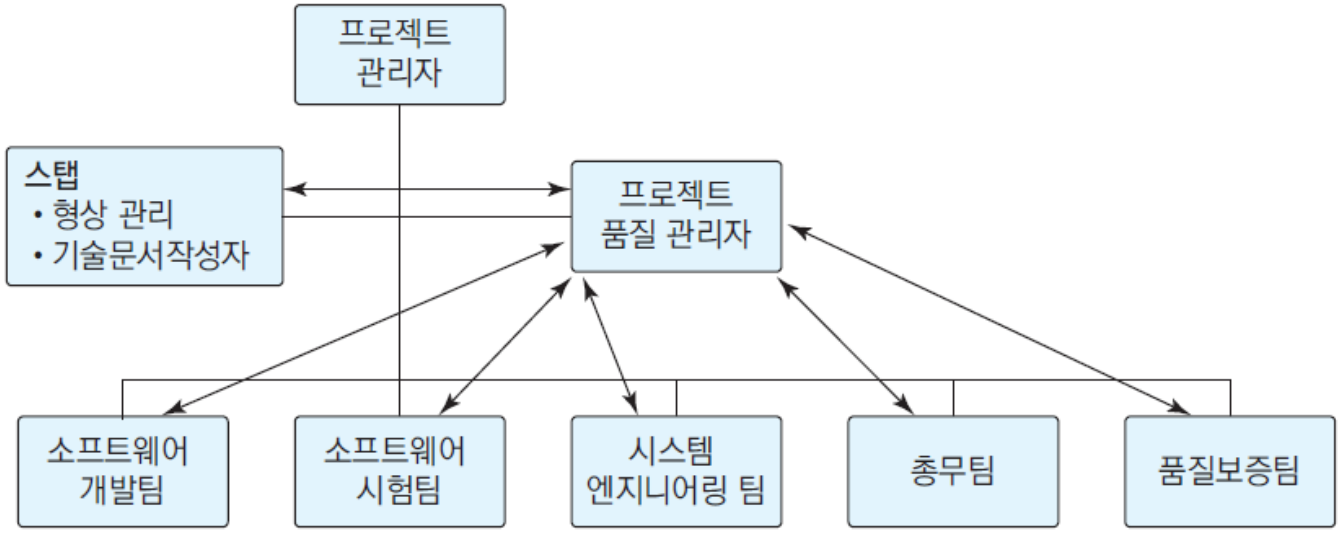


그림 12.10 품질 관리 조직

프로세스와 표준을 정의

- 소프트웨어 개발, 품질 관리 프로세스 및 방법론의 정의
- 개발 주기 동안 품질보증 작업을 수행할 표준, 절차, 가이드라인의 정의
- 품질 측정과 평가를 위한 품질 메트릭, 지표 정의

전통적인 프로세스의 품질보증

표 12.1 전통적인 프로세스의 품질보증

소프트웨어 개발 작업	전통적인 프로세스의 품질보증 활동
소프트웨어 요구 분석	1. 요구 명세 검토 2. 요구와 관련된 메트릭과 지표 평가 3. 프로토타이핑 4. 모델 검토 5. 소프트웨어 인수 시험 계획
소프트웨어 설계	1. 소프트웨어 설계 리뷰, 인스펙션, 워크스루 2. 설계 관련 메트릭과 지표 평가 3. 유스케이스 중심 검토 4. 모델 체킹 5. 소프트웨어 통합 테스팅 계획
소프트웨어 구현	1. 코드 리뷰, 인스펙션, 워크스루 2. 정적 분석 체킹 3. 메트릭과 지표에 근거한 평가 4. 코딩 표준
통합 및 시스템 테스팅	1. 통합 테스팅 2. 인수 테스팅
소프트웨어 운용 및 유지보수	1. 변경 분석 및 제어 2. 소프트웨어 리엔지니어링 3. 리그레션 테스팅

품질 보증 활동

1. 품질 계획 – 프로젝트 초반에 이루어지는 활동. 특정 프로젝트에 대한 품질 계획을 작성한다.
2. 품질 제어 – 프로젝트 전반에 걸쳐 이루어짐. 품질 계획의 실행을 모니터링하고 현실적으로 수정이 필요하면 품질 계획을 수정

품질관리

- 품질계획
  - 각 프로젝트 초기에 이루어짐
  - 목적
  - 관리
  - 표준과 관례
  - 리뷰와 감리
  - 형상관리
  - 프로세스, 방법론, 도구, 기술
  - 메트릭, 지표
- 품질 보증 계획(IEEE730)

1. 계획의 목표
2. 관련 문서
3. 관 리
   - 3.1 조 직
   - 3.2 작 업
   - 3.3 책 임
4. 문서화
   - 4.1 목 적
   - 4.2 기본적으로 요구되는 문서
5. 표준, 실습, 관례, 메트릭
   - 5.1 목 적
   - 5.2 내 용
6. 검토와 검사
   - 6.1 목 적
   - 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

품질관리

- 품질보증 제어
  - 계획이 정확하게 실행되고 있는지 확인하는 것
  - 개발자가 품질보증 활동을 수행하도록 도와주는 것
  - 품질 관련 데이터를 모아 데이터베이스 사용하여 관리
  - 관리자에게 프로세스 개선을 위한 제안을 하고 수용된 제안이 적절히 실현되고 프로세스에 녹아 들었는지 확인

인스펙션

- 품질 보증을 위한 검토 작업

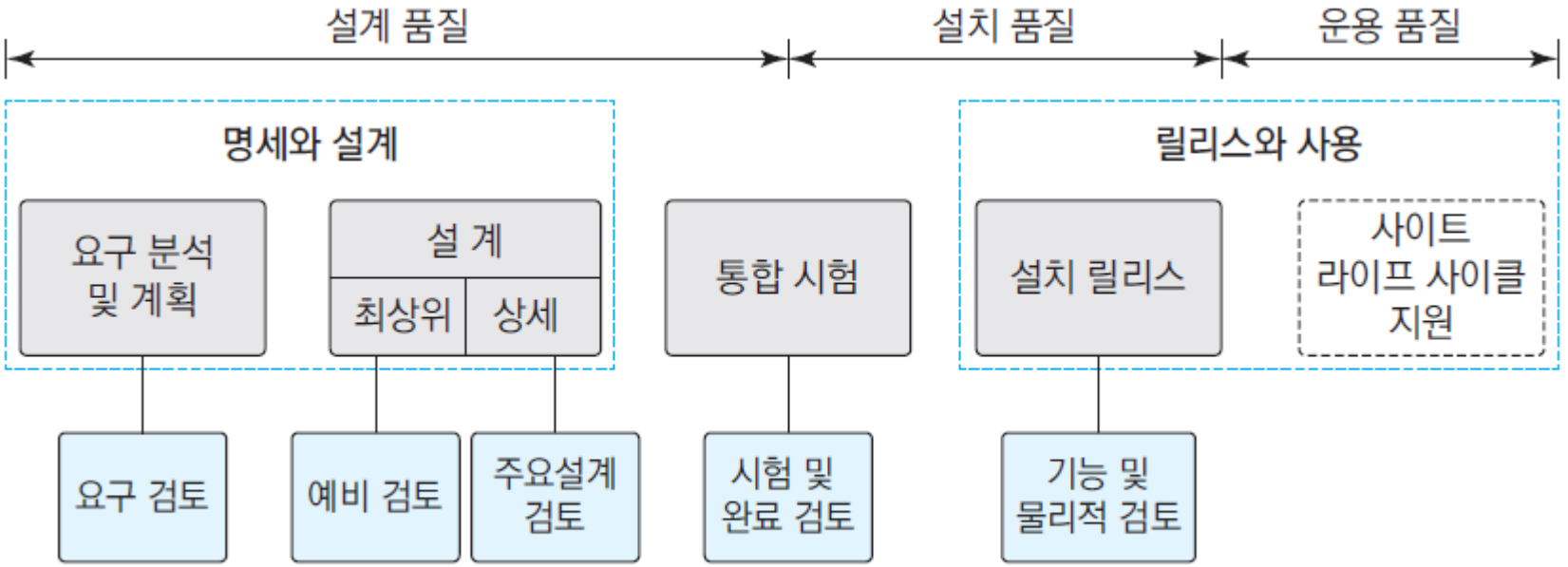
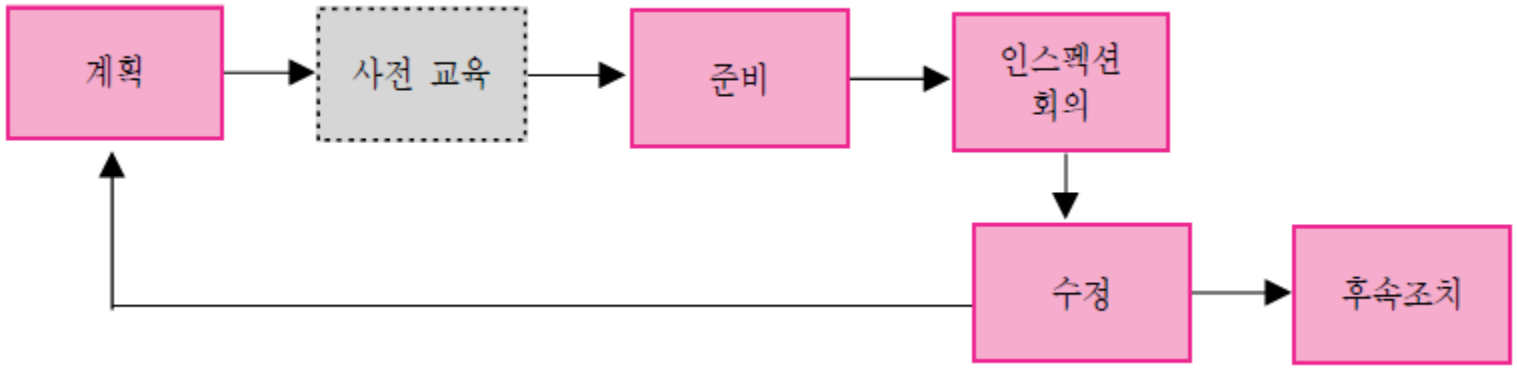


그림 12.11 품질 보증을 위한 검토 작업

- 결과물 검토 방법
  - 인스펙션, 워크스루, 동료 검토

인스펙션

- 품질 개선과 비용 절감을 위한 기법으로 사용
- 프로덕트를 공통되는 오류, 변칙, 표준이나 관례의 부적합 리스트와 체크해 보는 작업
- 인스펙션 과정



12.4 품질 측정

- 소프트웨어 측정(software measurement)
  - 소프트웨어 속성의 객관적이고 정량적인 평가

- 소프트웨어 메트릭(software metric)
  - 표준화된 소프트웨어 측정 방법

품질 측정과 메트릭의 유용성

- 품질 측정과 메트릭
  - 요구분석, 설계, 구현, 문서화가 포함된 소프트웨어의 정량적인 평가
- 지표의 정의와 사용
  - 지표 – 상대적 의미가 있는 값의 범위
- 중요한 부분에 자원을 투입
- 유사 프로젝트와 시스템을 정량적으로 비교
- 개선의 정량적인 평가
- 기술의 객관적인 평가
- 프로세스 개선의 객관적인 평가

전통적인 품질 메트릭

품질 메트릭	타입	설명	계산	
요구의 비모호성 (Q1)	R	총 요구의 수(n_r) 대비 검토인이 요구에 대하여 동일한 의미적 해석을 하는 요구의 개수 (n_{ui})	$Q1 = n_{ui}/n_r$	
요구의 완전성 (Q2)	R	상태와 자극의 모든 가능한 조합의 수($n_s \times n_i$) 대비 요구에 명시된 고유한 기능의 수(n_u)의 비율 n_s 는 상태의 수이며 n_i는 요구에 표시된 입력과 자극의 수	$Q2 = n_u/(n_s \times n_i)$	
요구의 정확성 (Q3)	R	요구 총 개수($n_r = n_c + n_{nv}$)대비 검증된 정확한 요구(n_c)의 비율. n_{nv}은 아직 정확한 요구로 검증되지 않는 요구의 수	$Q3 = n_c/n_r = n_c/(n_c + n_{nv})$	
요구의 일관성 (Q4)	R	요구 총 개수(n_r) 대비 상충되지 않는 해석을 가진 요구($n_r - n_{ci}$)의 비율. n_{ci} 는 검토인에 의하여 상충된 요구라고 알려진 요구의 개수	$Q4 = (n_r - n_{ci})/n_r$	
팬 인	D	호출한 모듈의 개수. 팬 인이 높은 모듈은 • 고장의 핵심 • 너무 많은 책임이 배정됨 • 변경되면 다른 많은 모듈에 영향을 줌		
팬 아웃	D	주어진 모듈에 의하여 호출된 모듈의 수. 팬 아웃이 너무 높으면 • 너무 많은 다른 모듈을 요구하기 때문에 재사용이 어려움 • 다른 모듈의 수정에 의하여 영향을 받음 • 다른 모듈과 상호작용 때문에 이해하기 어려움		

전통적인 품질 메트릭

품질 메트릭	타입	설명	계산	
요구의 비모호성 (Q1)	R	총 요구의 수(n_r) 대비 검토인이 요구에 대하여 동일한 의미적 해석을 하는 요구의 개수 (n_{ui})	$Q1 = n_{ui}/n_r$	
요구의 완전성 (Q2)	R	상태와 자극의 모든 가능한 조합의 수($n_s \times n_i$) 대비 요구에 명시된 고유한 기능의 수(n_u)의 비율 n_s 는 상태의 수이며 n_i는 요구에 표시된 입력과 자극의 수	$Q2 = n_u/(n_s \times n_i)$	
요구의 정확성 (Q3)	R	요구 충 개수($n_r = n_c + n_{nv}$)대비 검증된 정확한 요구(n_c)의 비율. n_{nv}은 아직 정확한 요구로 검증되지 않는 요구의 수	$Q3 = n_c/n_r = n_c/(n_c + n_{nv})$	
요구의 일관성 (Q4)	R	요구 충 개수(n_r) 대비 상충되지 않는 해석을 가진 요구($n_r - n_{ci}$)의 비율. n_{ci} 는 검토인에 의하여 상충된 요구라고 알려진 요구의 개수	$Q4 = (n_r - n_{ci})/n_r$	
팬 인	D	호출한 모듈의 개수. 팬 인이 높은 모듈은 • 고장의 핵심 • 너무 많은 책임이 배정됨 • 변경되면 다른 많은 모듈에 영향을 줌		
팬 아웃	D	주어진 모듈에 의하여 호출된 모듈의 수. 팬 아웃이 너무 높으면 • 너무 많은 다른 모듈을 요구하기 때문에 재사용이 어려움 • 다른 모듈의 수정에 의하여 영향을 받음 • 다른 모듈과 상호작용 때문에 이해하기 어려움		

전통적인 품질 메트릭

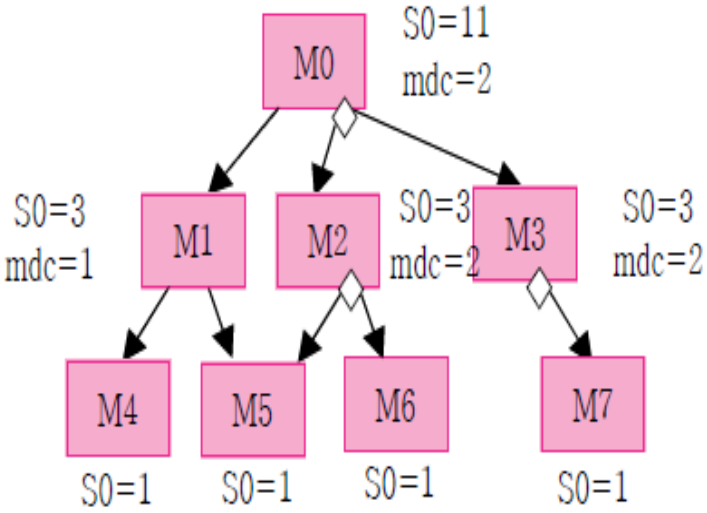
- 요구 메트릭(R)
  - 요구의 비모호성(unambiguity)
  - 요구 완전성 메트릭 – 요구 명세서에 기술된 시스템의 상태와 시스템에 대한 외부자극이 완전하다는 가정에 근거
  - SRS – 시스템의 모든 가능한 상태와 모든 가능한 외부자극을 포함

$$f(state, stimulus) \rightarrow (state, response)$$

  - f함수가 완벽하게 매핑된다면 SRS는 완벽한 것으로 간주

전통적인 품질 메트릭

- 설계 메트릭
  - M0-M7 – 모듈
  - 화살표 – 모듈 호출
  - 다이아몬드 화살표 – 분기 호출
- 모듈 설계 복잡도(mdc(M))
  - mdc(M) = d + 1
  - d : M이 가진 다이아몬드의 수
  - 다이아몬드 : 이진 조건의 분기
- 선택 복잡도(S0)
  - S0(leaf) = 1, 각 단말 노드는 하나의 서브트리
  - S0(M) = $\sum_{i-1}^{n} S0(M_i) + mdc(M)$



전통적인 품질 메트릭

- M4~M7의 S0 = 1
  - 모두 단말 노드
- M1 모듈복잡도
  - S0(M1) = S0(M4)+S0(M5)+mdc(M1) = 1+1+1 =3
    - M1은 분기가 없기 때문에 mdc 값이 1
- 모듈 설계 복잡도
  - S0(M) = N_{dm}+N_{adb}
  - N_{dm} : 모듈의 개수
  - N_{adb} : 선택적 모듈을 호출하는 분기의 수

전통적인 품질 메트릭

- 구현 및 시스템 메트릭
- 구현 메트릭
  - LOC 메트릭 : 원시코드의 줄을 세는 것
  - 싸이클로매틱 복잡도 메트릭 : 프로그램을 통과하는 독립된 경로의 개수이며 필요한 테스트의 횟수
- 시스템 메트릭
  - 신뢰도 메트릭
    - MTBF = MTTF + MTTR
    - MTBF(Mean Time Between Failure) : 고장 사이의 평균 시간
    - MTTF(Mean Time To Failure) : 고장 까지의 평균시간
    - MTTR(Mean Time To Repair) : 수리 평균시간

12.5 프로세스 개선

- 엔지니어링 프로세스가 경험에 따라 어떤 차이가 있는지 연구하고 모델 제시
  - 미 국방성의 CMMi
  - ISO의 SPICE
- 소프트웨어 시스템의 품질은 그것을 개발하는데 사용되는 프로세스의 품질에 좌우됨

CMMi

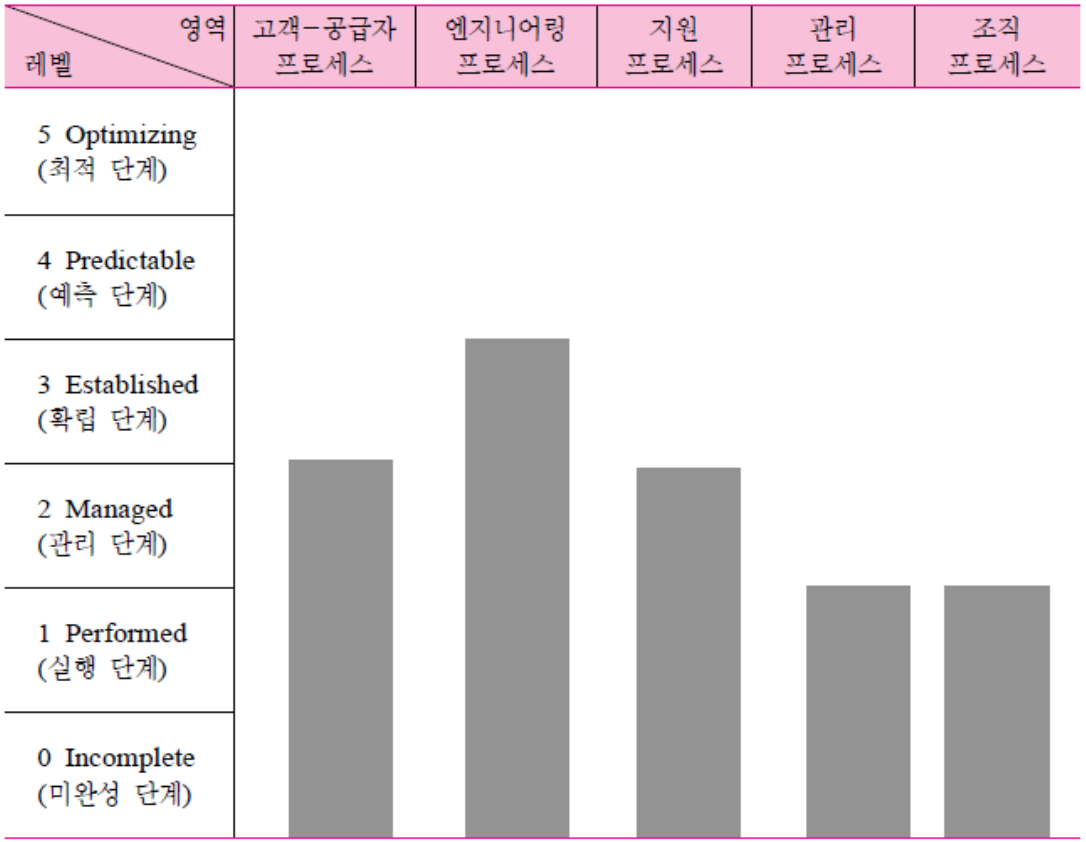
- 프로세스 성숙도를 위한 프레임워크
  - CMM-SW: 소프트웨어 개발 프로세스의 성숙도를 다룸
  - CMMi는 소프트웨어, 시스템, 프로덕트를 포함하는 세 분야를 통합 평가하는 모델
- 용도
  - 성숙도 평가 기준
  - 능력을 스스로 평가하고 개선의 방향을 설정

CMMi 모델

레 벨	초 점	주요 프로세스 영역	결과
5 Optimizing (최적단계)	계속적인 개선	• 인과관계 분석 및 해결(CAR) • 조직성과 관리(OPM)	생산성과 품질
4 Quantitatively Managed (계량적 관리 단계)	프로덕트 및 프로세스 품질	• 조직 프로세스 성과(OPP) • 계량적 프로세스 관리(QPM)	
3 Defined (정의 단계)	엔지니어링 프로세스	• 조직 프로세스 초점(OPF) • 조직 프로세스 정의(OPD) • 조직 교육 훈련(OT) • 통합 프로젝트 관리(IPM) • 위험 관리(RSKM) • 요구사항 개발(RD) • 기술 솔루션(TS) • 제품 통합(PI) • 검증(VER) • 확인(VAL) • 의사결정 분석 및 해결(DAR)	
2 Managed (관리 단계)	프로젝트 관리	• 프로젝트 계획(PP) • 프로젝트 모니터링 및 통제(PMC) • 형상관리(CM) • 측정 및 분석(MA) • 프로세스 및 제품품질 보증(PPQA) • 요구 사항 관리(REQM) • 공급자 협약 관리(SAM)	
1 Initial (초보단계)	영웅적 개인		위 험

ISO 9001

- SPICE(Software Process Improvement and Capability dEtermination)
  - 소프트웨어 프로세스 평가를 위한 구제 표준



CMMi와 SPICE

- 차이점은 성숙도 레벨과 심사 영역의 구분
- 성숙도
  - CMMi는 레벨 1부터 5까지 5개의 성숙도 수준
  - SPICE는 레벨 0부터 5까지 6개의 수준
- 심사 영역
  - CMMi는 하나의 레벨로 평가하는 일차원적인 구조
  - SPICE는 각 프로세스 영역마다 능력에 대한 평가를 별도로 하는 이차원적 구조