10/9 보강 : 12/7 까지 스냅보드 강의

13장. SW품질

숙명여자대학교
SOOKMYUNG WOMEN'S UNIVERSITY

Contents

1. 소프트웨어 품질 개요 (1/3)

주관적 개념

↳ 향상; 사용자+SW개발관련자모두를 만족시킴

- 어떤 스마트폰을 선택할 것인가?
  - 선택의 기준은 무엇인가?
  - 사용자는 무엇을 기대하고 있는가?

미래의 스마트폰

1. 소프트웨어 품질 개요 (2/3)

- 소프트웨어 품질의 특징
  - 눈으로 확인할 수 없음
  - 개발 초기에 사용자의 요구를 정확히 알 수 없음
  - 시간이 지날수록 사용자가 원하는 품질 수준이 점점 높아짐 ∴ 기술이 빨리변함
  - 100점, 90점과 같이 절대적으로 평가할 수 없음
  - 품질은 다양한 관련자들을 고려해야 하는 다차원적인 것
    → 한가지로 정의할수 없음

1. 소프트웨어 품질 개요 (3/3)

- 소프트웨어 품질의 정의
  - 1990년 IEEE
    - 시스템, 구성 요소 또는 프로세스에 명시된 요구사항을 충족시키는 정도 → 고객 및 사용자의 요구 및 기대를 충족시키는 정도
  - 1999년 ISO
    - 요구사항을 만족하는 소프트웨어 제품의 능력
  - 2004년 프레스만(Pressman)
    - 명시적인 기능 및 성능 요구사항, 명시적으로 문서화된 개발 표준, 개발된 소프트웨어에서 기대되는 묵시적인 특성에 대한 적합성

1.1 SW 품질의 중요성

=품질속성

- 소프트웨어 품질 요소는 매우 다양
- 이 모든 요소를 모두 만족시키지 못하기 때문에 대상 소프트웨어의 특성을 고려한 적절한 품질 목표 설정 중요

적절한!

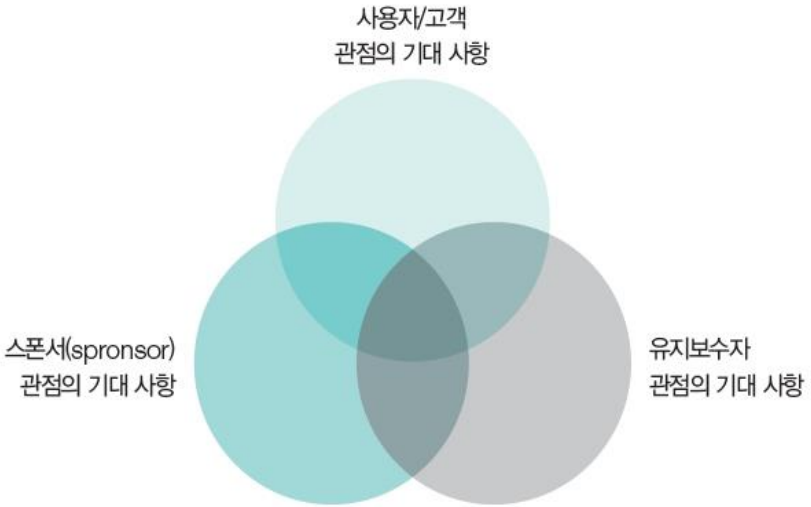
소프트웨어 품질 특성을 나타내는 품질 아이스버그

SW제품 자체에 대한

1.2 소프트웨어 스테이크 홀더

= SW개발에 관련된 이해당사자

- 주요 스테이크홀더



기대치가 다름

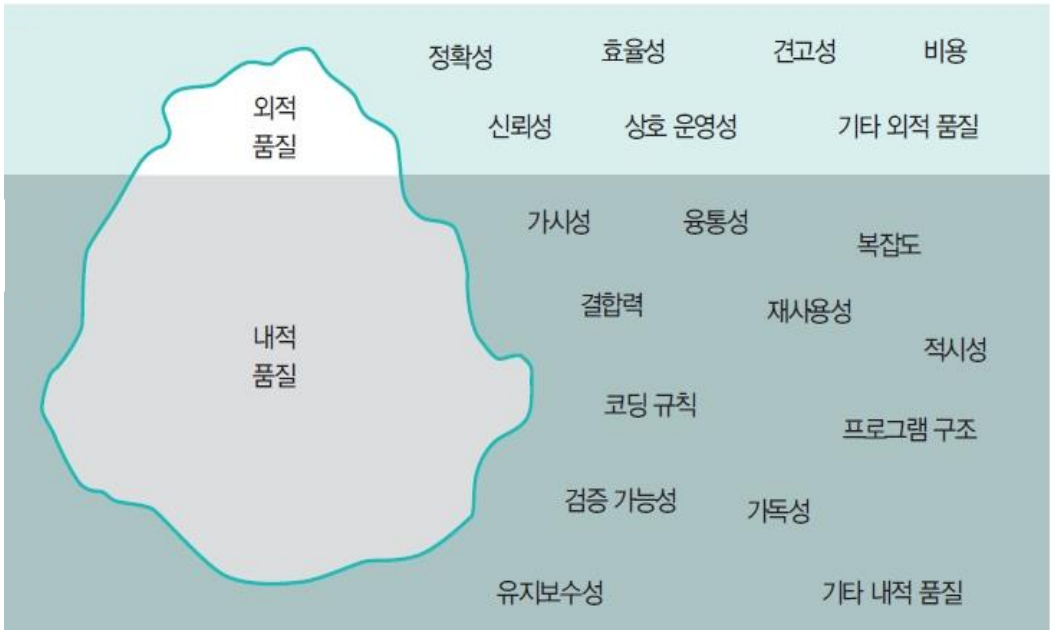
역할	기대치	대응 품질 요소
스폰서 (Sponsor)	• 적은 비용으로 소프트웨어가 개발되기를 원한다. • 개발된 소프트웨어가 다른 응용에 사용되기를 원한다. • 기존 컴포넌트의 재사용을 통해 개발되기를 원한다. • 비즈니스 과정에 이득이 되기를 원한다.	- 낮은 비용 - 적용성(Adaptabiliy) - 재사용성 - 비용 효율성
사용자 (User)	• 소프트웨어의 기능이 정확히 동작하기를 원한다. • 고장이 나지 않기를 바란다. • 사용하기 쉬워야 한다. • 적은 비용으로 구매하기를 원한다.	- 정확성 - 신뢰성 - 사용성(Usability) - 낮은 비용
유지보수자 (Maintainer)	• 소스 코드를 이해하기 쉬워야 한다. • 표준 코딩 스타일에 따라 코드가 개발되기를 원한다. • 변경 영향이 한 부분으로 국한되기를 원한다. • 수정된 코드를 쉽게 테스트하기를 원한다. • 코드와 일관성 있는 문서가 제공되어야 한다.	- 가동성 - 코딩 표준 준수성 - 프로그램 구조 - 검증 가능성 - 문서화

2. 소프트웨어 품질 요소

제품자체 — 외적 ; 사용자 / 내적 ; 개발자

프로세스

- 외적 품질 요소(External Quality)
  - 기능적인 품질 : SW가 기대되는 동작을 하는지 등
- 내적 품질 요소(Internal Quality)
  - 구조적인 품질. 내적품질이 확보되지 않으면 외적 품질도 낮아지게 됨
- 프로세스 품질



2.1 SW 외적 품질 요소 (1/5)

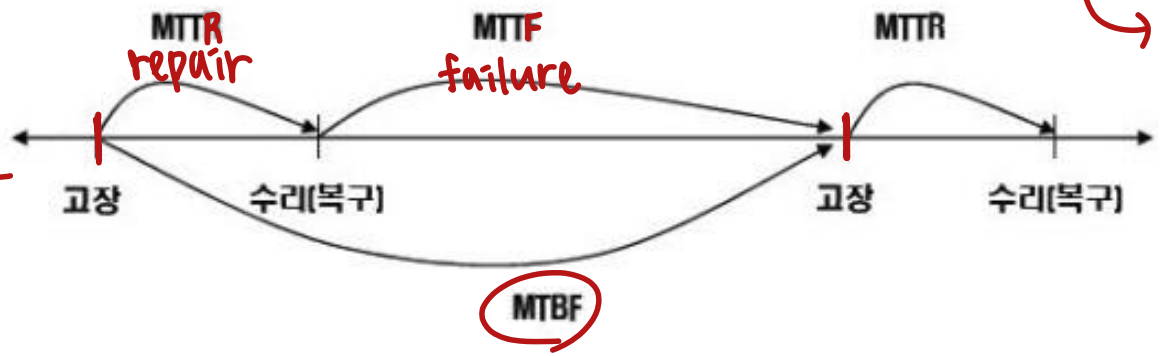
사용자관점

1. 정확성(Correctness)
   - 주어진 명세서의 내용을 하나씩 테스트하여 원하는 결과를 생성하는지 여부로 판단
   - Correctness(P) = 1 – (A/B) = 1 - 실패한 요구사항 / 전체 요구사항
     - P : 정확성을 알고 싶은 프로그램
     - B : 사용자의 전체 요구사항 개수
     - A : 테스트를 통해서 명세 내용대로 동작하지 못한 기능 수

2.1 SW 외적 품질 요소 (2/5)

- 신뢰성(Reliability)
  - 소프트웨어를 사용하는 동안 나타나는 오류(정확하게는 고장) 발생 정도로 판단
  - 소프트웨어 고장의 빈도수와 그 치명도로 나타냄
  - 예) MTBF (Mean Time Between Failure) = 10,000Hour

① 동작안됨
② 올바르지않은결과



고장과고장사이의평균시간
: 이시간만큼 고장없이 쓸수있음

2.1 SW 외적 품질 요소 (3/5)

4. 견고성(Robustness) ← 신뢰성보다 엄격함 ; 안전이 담보돼야하는 시스템 (ex. 항공기 제어)
   - 사용자가 제시한 요구사항 명세에 정의하지 않은 조건이나 환경에서도 소프트웨어가 합리적으로 동작해야 견고 ex) 아이디가 사용가능한 문자의 조합인지? → 이에 따른 시스템의 응답
4. 성능(Performance)
   - 소프트웨어의 효율성(Efficiency)을 의미
   - 소프트웨어를 수행하기 위해 필요한 메모리의 양(Byte), 총 실행 시간(μsec) 등이 척도

→ 개발 이후에만 측정 가능 ; 시뮬레이션, 코드분석 등으로 예측

2.1 SW 외적 품질 요소 (4/5)

5. 사용자 친숙성(User Friendliness)
   - 소프트웨어가 사용하기 편리한 정도 → 인터페이스관점
   - 편의성 지원 기능이 얼마나 제공되는가로 측정
     - UI에서 도움말 말풍선, 스크롤 바, 핫 키 등의 개수
6. 가용성(Availability) 사용하고싶을때 언제든지!
   - 서버와 네트워크, 프로그램 등의 정보 시스템이 정상적으로 사용 가능한 정도
   - 예)

$$\text{가용성} = \frac{\text{Usable Time}}{\text{Operation Time}} = \frac{\text{사용가능시간}}{\text{운용시간}}$$

2.1 SW 외적 품질 요소 (5/5)

⑦ 보안성(Security)

- 외부의 악의적인 공격이나 해커(Hacker)의 위협을 소프트웨어가 막아낼 수 있도록 구현하여 잠재적인 공격이 예측되는 상황에서도 소프트웨어가 올바르게 동작
- 소프트웨어 시스템에서 발견된 취약점의 개수, 사고 통계, 보안 취약으로 인한 연간 손실액 등이 척도

⑧ 안전성(Safety), ⑨ 무결성(Integrity) 등

↳ 위험상태를되도록 피하고 위험상태인경우 사용자에게 알림

↳ 허용되지않은접근막기 ; 데이터손실을막음

다익비다트 : SW 오작동 → HW 제어불가 → 사고원인

2.2 SW 내적 품질 요소 (1/5)

개발자 관점

- 검증 가능성(Verifiability)
  - 소프트웨어가 지닌 속성이 올바르다는 것을 안전하게 확인 가능
  - 정형 검증(Formal Verification)과 테스트로 평가
    - 정형 검증 : 검증 대상을 형식 언어로 표현하고 이를 풀거나 시뮬레이션함으로써 검증 대상이 정확하게 동작한다는 것을 보이는 방법
    - 테스트 : 소프트웨어의 정확한 동작을 확인하기 위하여 적절하게 생성된 테스트 데이터를 이용하여 실행시키는 방법

정형언어

수학적방법이론

자동화된도구사용

⇒ 여러기법혼합사용, 테스트로확인, …

2.2 SW 내적 품질 요소 (2/5)

↪ 얼마나 쉽게 할 수 있느냐 : 수정가능성으로 평가 ⇒ 발견된 오류, 결함을 수정하는데 들어가는 시간으로 평가

② 유지보수성(Maintainability)

- 수정 유지보수(Corrective Maintenance)
- 적응 유지보수(Adaptive Maintenance)
- 완전 유지보수(Perfective Maintenance)
- 예방 유지보수(Preventive Maintenance)

④ 이식성(Portability) ⇒ 지원가능한 플랫폼의 개수로 평가

- 얼마나 다양한 하드웨어 플랫폼을 지원하는가와 얼마나 다양한 버전의 소프트웨어 플랫폼을 지원하는가

2.2 SW 내적 품질 요소 (3/5)

Include 라이브러리 이클립스 플러그인 오픈라소 ⋮

④ 재사용성(Reusability)

- 새로운 소프트웨어를 개발하기 위해 기존 소프트웨어 컴포넌트를 사용하는 정도

- Reusability(P) = LOC(R)/LOC(S) = 재사용한 양 / 전체 개발
  - P : 특정 대상
  - LOC(S) : 전체 개발된 산출물의 양
  - LOC(R) : 재사용에 의해 개발된 부분의 양

분석·설계 단계에서도 재사용 有 ⇉ 재사용 컴포넌트 양 / 전체 개발 문서 양

2.2 SW 내적 품질 요소 (4/5)

생산성평가가 쉽지 않아서 외적품질요소를 사용해 평가 ⇒ 단위시간당 개발량(LOC) function point 이용

⑤ 생산성(Productivity)
- 외적 품질 요소인 성능(Performance)의 속성을 적용한 내적 품질 속성
- 주어진 시간 내에 얼마만큼의 성과를 내고 있는가를 척도로 사용

⑥ 상호 운용성(Interoperability)
- 서로 다른 소프트웨어들이 협업을 수행할 수 있는 능력을 충분히 제공하는 것
- IoT(Internet of Things) 기술을 근간으로 하는 스마트 시티 환경에서 매우 중요 → 인터페이스, 통신프로토콜등의 표준화 ※ 유비쿼터스 환경

2.2 SW 내적 품질 요소 (5/5)

- 기타 품질 요소
  - ⑦ 적시성(Timeliness)
  - ⑧ 가시성(Visibility)
  - ⑨ 회복성 (Recoverability)
  - ⑩ 변경성(Changeability)
  - ⑪ 적응성 (Adaptability)
  - ⑫ 추적성(Traceability) 등
  - ⋮

2.3 프로세스 품질

SW개발과정에 대한 품질

- 프로세스 품질개선 → product 자체에 영향을 준다!
  - 프로세스 관련 표준 : ISO 12207, CMMI, ISO 15504:SPICE
- 엔지니어링 관점의 프로세스 품질 요소
  1. 프로세스 모델 적합성
  2. 개발 방법론 적합성 (구조적, 객체지향, …)
  3. 도구 적합성 : CASE(Computer-Aided SE) 도구들 (→ 개발환경, 기술수준이 적합한지)
  4. 표준 준수성 (웹, 코딩, 프로토콜, 문서화… → 적절한지 & 준수해서 진행되고있는지)
  5. 프로젝트 데이터 관리 수준 → 프로젝트내에서 / 유사한 프로젝트진행시 활용할 수 있도록 (소스코드, 개발비용, 일정, 인력, …)

⇒ "가이드라인"

3. AI 소프트웨어 품질 (1/2)

- 전통적인 소프트웨어 vs. 인공지능 소프트웨어
  - 학습 모델의 입력과 결과 사이의 관계는 입력 데이터의 일부에 대해서만 정의 됨
  - 캡슐화 및 모듈화 같은 소프트웨어 엔지니어링의 일반적인 개발 원칙을 그대로 적용하기 어려움
  - 기계학습을 포함하는 컴포넌트의 개발과 통합의 접근 방법은 매우 다양한 방식에 의해 이루어질 수 있음
  - 학습 및 테스트에 사용하는 데이터가 알고리즘 보다 훨씬 더 중요하게 고려 됨

3. AI 소프트웨어 품질 (2/2)

- AI SW 품질에 대한 연구
  - EU : Ethics Guidelines for Trustworthy
  - 독일 DIN SPEC 92001
  - 일본 QA4AI Consortium
- AI SW 품질속성
  - 투명성과 책임
  - 다양성/공정성 및 사회적 웰빙
  - 보안과 안전성
  - 기술적 견고성 및 신뢰성
  - 법적/윤리적 측면

3.1 AI SW 품질 특성 (1/4)

지금의 AI는 블랙박스 → 정확도(누치)로 평가, 근거는 모름

1. 투명성(Transparency)과 책임(Accountability) ← 누가에게 책임이 있는지 설명가능해야됨 / ↙ 필요!
   - 기계학습 기반 시스템은 동일한 입력에 대하여 서로 다른 결과를 제공할 수 있음 → 재현성을 보장할 수 있는 방향으로 개발하도록
   - AI 시스템이 제공하는 출력은 시간에 따라 변할 수 있다는 가능성 때문에, 출력에 대한 해석 가능성 및 설명 가능성이 중요 => XAI(eXplainable AI)
   - 출력 결과가 부정적인 결과를 유발하는 경우, 이를 사용자에게 알려주는 보고 기능이 부가적으로 제공되는 것이 좋음

안심하고 사용할 수 있도록

3.1 AI SW 품질 특성 (2/4)

② 다양성(Diversity), 공정성(Fairness), 사회적 웰빙(well-being)

- AI 시스템 개발에서 학습 모델의 구성과 이에 필요한 의사결정 정책 등은 특정 요소에 편중하여 출력을 제공하는 오류가 있어서는 안됨 입력데이터의 편향성↓
- 모든 가능한 데이터들을 수용하여 출력을 제공할 수 있는 모델 설계 필요
- 다양한 이해관계자의 참여를 통해 AI 시스템은 환경 친화적 방향으로 구축되어 사회적 웰빙을 제공해야 함

3.1 AI SW 품질 특성 (3/4)

3. 보안(Security)과 안전성(Safety)
   - AI 시스템의 응용 범위가 넓어져 개인 정보의 누출이나 프라이버시 침해 소지가 있음
   - 정확한 문제 해결을 위해서 사용되는 데이터가 좋은 품질과 무결성을 가져야 하는 것은 물론, 데이터에 존재할 수 있는 오류, 노이지(Noisy), 정체 불명 또는 악의적인 데이터들에 대처할 수 있어야 함
4. 기술적 견고성(Robustness)과 신뢰성(Reliability)
   - 유해한 입력, 오류 있는 입력에 대해서도 믿을 만하고 이해할 만한 결과가 제공되어야 함

3.1 AI SW 품질 특성 (4/4)

법적·윤리적 측면

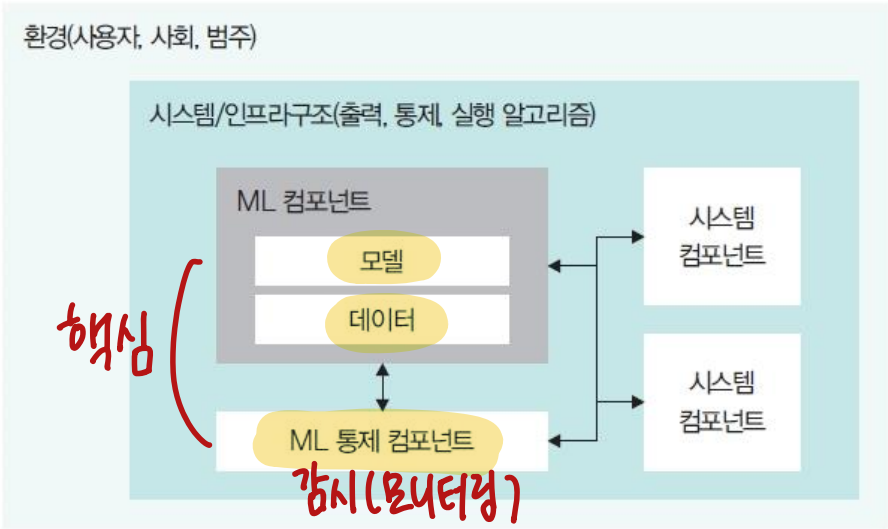
: 시스템의 잘못인가 사람의 잘못인가

- AI 시스템이 인간이 해온 일들을 대행하는 에이전트(Agent) 역할을 수행하거나 인간이 하는 일들을 감독하고 모니터링할 수 있으므로 인간에게 적용되는 법적· 윤리적 문제들을 보장할 수 있도록 개발되어야 함

↳ AI SW에도 똑같이 적용

3.2 시스템 관점별 품질요소 (1/2)

- ML 시스템의 구성 관점



기계학습 기반 소프트웨어 시스템의 구성 관점

2020년 독일의 프란호퍼 연구소에서 제시한 모델

1) 모델 관점
   - 학습 모델과 관련되며, 분류나 차원 축소 같은 작업을 수행하기 위해 데이터에 대한 훈련이 이루어지는 부분

3.2 시스템 관점별 품질요소 (2/2)

2) • 데이터 관점
  - 모델에 입력되는 실데이터와 관련된 부분 데이터품질

3) • 시스템 관점
  - 기계학습 컴포넌트(모델과 데이터)들을 연결하고, 형상(Configuration)을 정의하는 부분

4) • 인프라 관점
  - 어떻게 구현되는가에 초점을 맞춘 부분으로 시스템 관점과 밀접한 관계가 있음

5) • 환경 관점
  - 시스템 외부 관점으로 시스템과 사용자가 어떻게 상호작용 하는가를 표현하는 부분

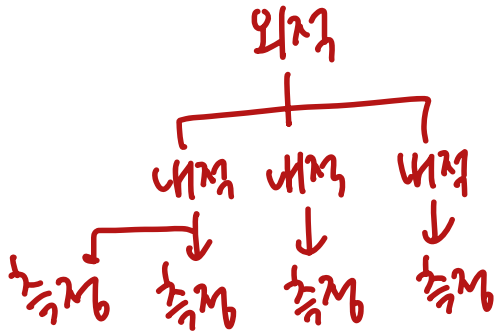
⇒ SW 품질에서 이해관계자(stakeholder) 마다 관점·품질기대치가 다른것처럼!

4. 소프트웨어 품질 모델 및 표준

- McCall의 FCM
- HP의 FURPS 모델
- ISO 9126 품질 모델
- ISO 25010 품질 모델

↑ 국제표준

4.1 McCall의 FCM 모델 (1/2)



- FCM의 기본 개념
  - Factors
    - 사용자에게 보이는 소프트웨어의 외적 특성을 기술하며, 시스템의 동작 특성을 나타냄
  - Criteria
    - 개발자에게 보이는 소프트웨어의 내적 특성을 기술하며, 소프트웨어 개발 및 설계와 관련된 품질 요소를 정의
  - Metrics
    - 소프트웨어의 내적·외적 특성을 측정하기 위한 기본 단위와 방법을 정의하고 설명

4.1 McCall의 FCM 모델 (2/2)

- 운영관점, 개선 관점, 전환 관점으로 구분

세부품질요소의

구분	세부 품질 요소
① 제품 운영	정확성(Correctness), 신뢰성(Reliability), 효율성(Efficiency), 무결성(Integrity), 사용성(Usability)
② 제품 개선	시험 가능성(Testability), 융통성(Flexibility), 유지보수성(Maintainability)
③ 제품 전환	이식성(Portability), 재사용성(Reusability), 상호 운용성(Interoperability)

1995 캘러헤인

4.2 HP의 FURPS 모델

→ FURPS+ 모델 : 생산라인에서 제품품질을 향상시키기위해 적용

- F (Functionality) 기능
  - SW 수행기능, 기능의 일반성 및 보안성 요소들
- U (Usability) UI
  - SW 외관구성 및 특성, 일관성 및 문서화 요소들
- R (Reliability)
  - 고장빈도, 치명도, MTBF, 고장회복력 등
- P (Performance) 성능
  - 처리속도 및 응답시간, 자원사용율 등
- S (Supportability) 지원가능성
  - SW 확장, 적용, 수정 등과 관련된 요소들

4.3 ISO 9126 품질 모델 (1/2)

- ISO/IEC 9126: 2001 국제표준
  - Software Engineering : Product Quality

4.3 ISO 9126 품질 모델 (2/2)

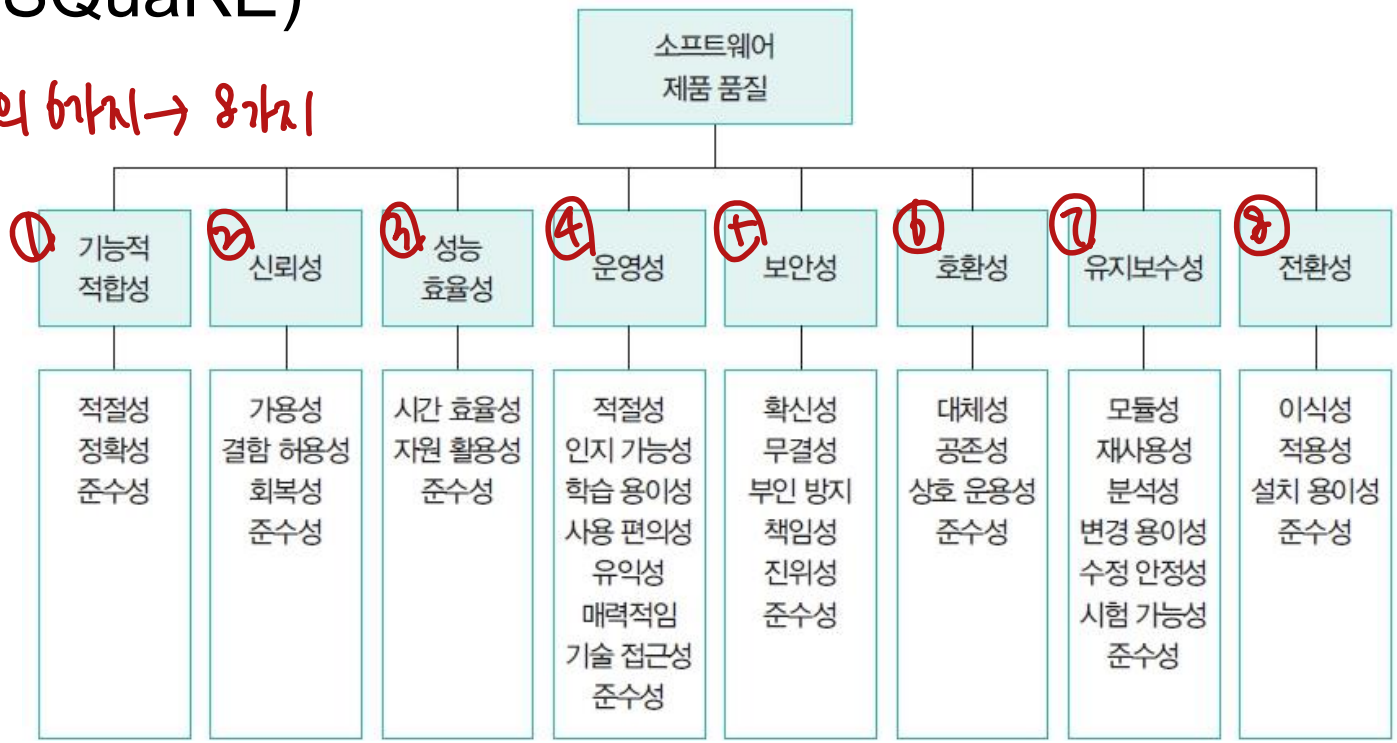
- ISO 9126에서 정의한 세부품질 요소 + 측정척도 metrics까지정의

품질 특성	세부 품질 요소
① 기능성	적합성(Suitability), 정확성(Accuracy), 상호 운용성(Interoperability), 준수성(Compliance), 보안성(Security)
② 신뢰성	성숙성(Maturity), 결함 허용성(Fault-Tolerance), 회복성(Recoverability)
③ 사용성	이해성(Understandability), 학습 용이성(Learnability), 운영성(Operability)
④ 효율성	시간 효율성(Time behaviour), 자원 효율성(Resource behaviour)
⑤ 유지보수성	분석성(Analyzability), 변경성(Changeability), 안정성(Stability), 시험 가능성(Testability)
⑥ 이식성	적용성(Adaptability), 설치 용이성(Installability), 부합성(Conformance), 대체 가능성(Replaceability)

4.4 ISO 25010 품질 모델

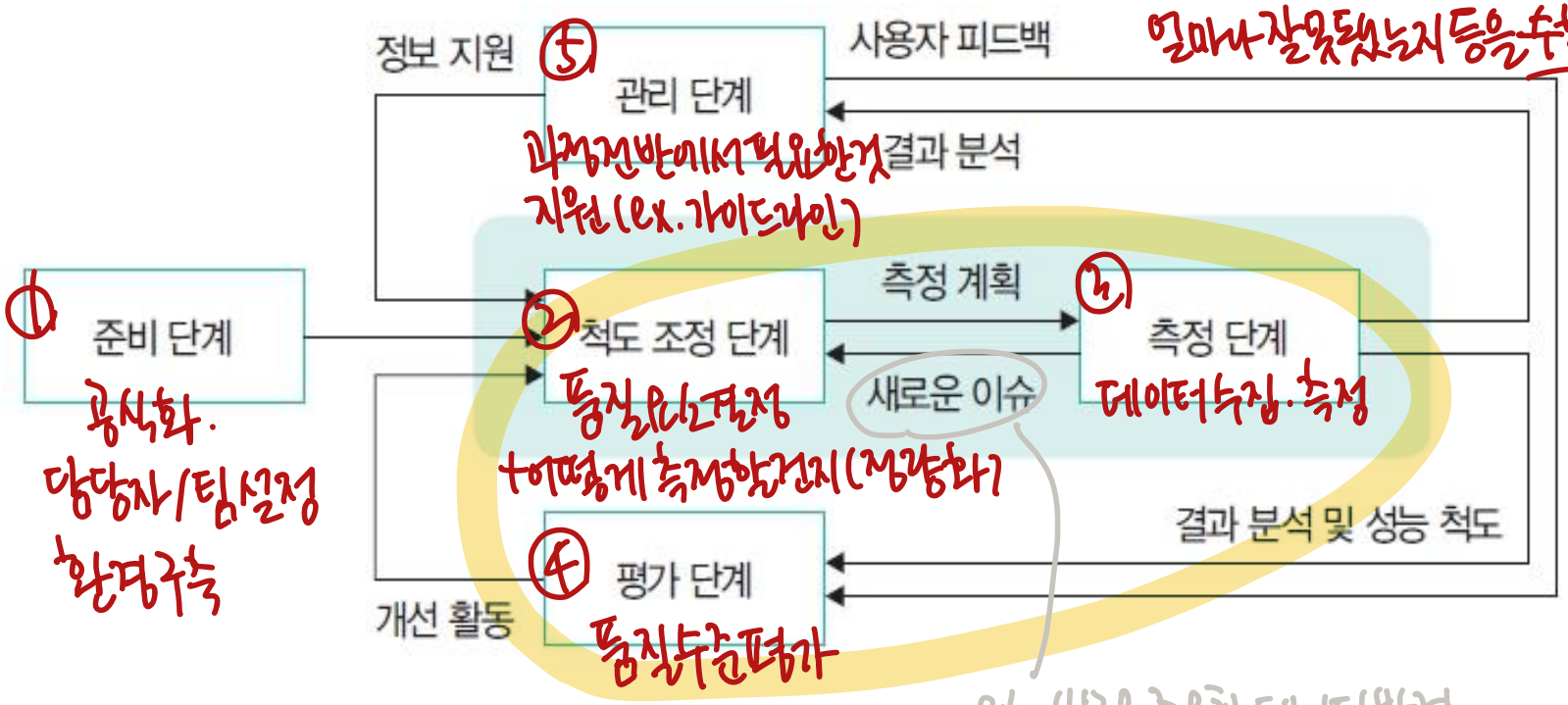
- ISO/IEC 25010: 2011 국제표준 (9126을대체)
  - Systems and software engineering-Systems and software Quality Requirements and Evaluation (SQuaRE)

9126의 6가지 → 8가지



5. 소프트웨어 품질 개선

- 소프트웨어 개발 과정에서 적용하는 정량적 품질 관리 프로세스(5단계)



관리하기위해서 필수! 얼마나잘됐는지등을수치화해야함

과정전반에서 필요한것 지원(ex.가이드라인)

공식화. 담당자/팀설정 환경구축

품질목표설정 +어떻게 측정할것지(정량화)

데이터수집.측정

ex) 새로운중요한데이터발견 척도가 잘못됨

10/9