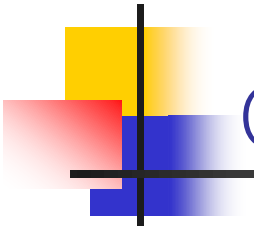
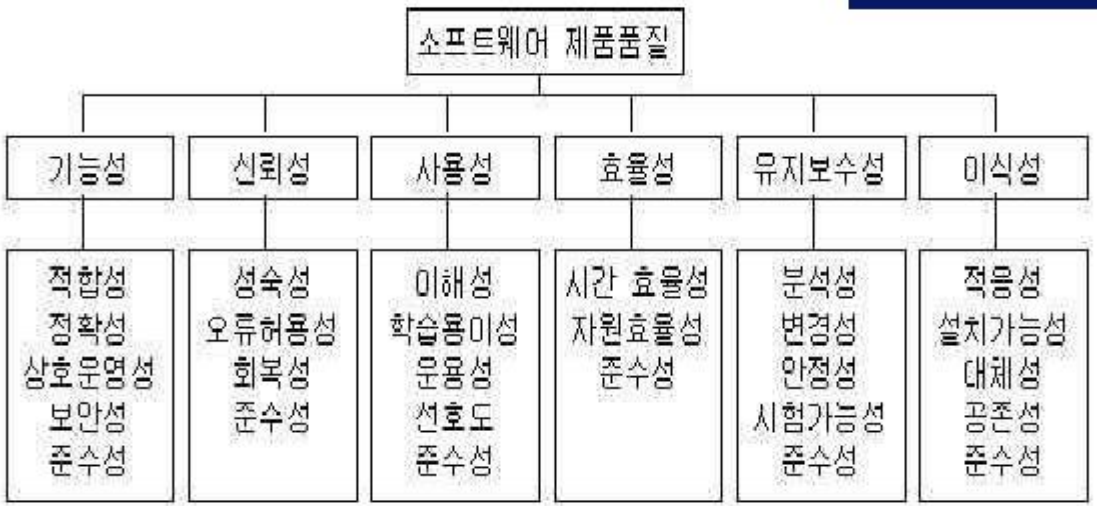
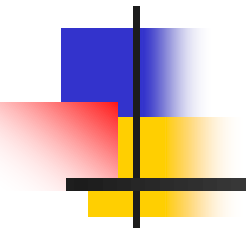


Chapter 13. 정리

최적화

품질은
사람에 의해서 결정된다





제 19장. 품질

May. 2018
Young-gon, Kim
ykkim@kpu.ac.kr
Department of Computer Engineering
Korea Polytechnic University

Topics covered

- 품질
- 소프트웨어 품질
- 소프트웨어 품질 딜레마
- 소프트웨어 품질 확립
- 양질의 소프트웨어
- 품질 속성 이해

1. 품질

◆ 관점별 품질

- **초월적**(한계를 벗어남) 견해: 당신이 즉시 인식할 수 있는 것이지만, 명백하게 정의할 수 없는 것
- **사용자** 견해 : 최종사용자 구체적인 목표-> 목표 충족
- **제조자** 견해 : 제품의 원래 사양 관점
- **제품**의 관점 : 제품의 고유한 특성(기능과 특성)
- **가치 기반**의 관점 : 고객이 제품을 사기 위해 얼마나 지불할 의사가 있는지

◆ 품질=사용자 만족

- 구현 설계에 따른 실행과 초래된 시스템이 그 요구사항 및 성능 목표를 충족 하는 정도
- 사용자 만족 = **대응 상품 + 좋은 품질 + 예산 계획 내 납품.**

2.소프트웨어 품질

◆ 소프트웨어 품질

- 생산하는 **사람** + 소프트웨어 **프로세스**

◆ 소프트웨어 품질의 중요한 포인트

- 효과적인 **소프트웨어 프로세스**
  - 보호 활동도 품질에 중요 : 변경관리 및 기술 검토
- **고품질의 소프트웨어**
  - 유용한 제품 : 최종 사용자가 원하는 컨텐츠, 기능, 특징 제공
  - 일련의 암시적인 요구사항 만족 : 사용 용이성
- 소프트웨어 제품의 **제조사와 사용자 모두에 가치**를 둠
  - 적어진 유지보수 노력
  - 적은 버그 수정
  - 감소된 고객 지원 필요.

2.소프트웨어 품질

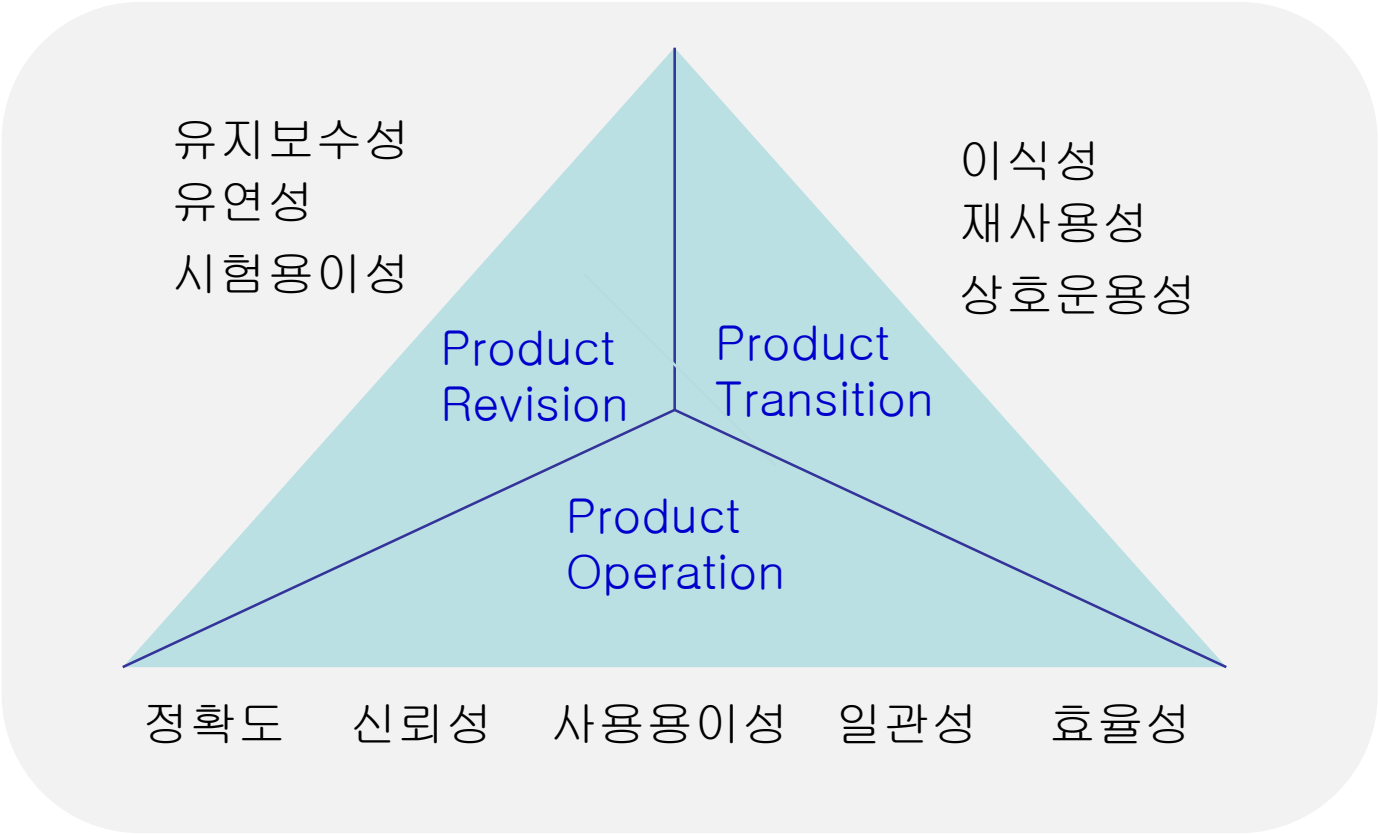
품질 치수 : 소프트웨어 품질 고려시 적용

- **성능 품질**
  - 최종 사용자에게 가치를 제공하는 방법, 모든 콘텐츠/기능 제공하는가 ?
- **기능의 품질**
  - 처음 최종사용자를 놀라게 하고 기쁘게 할 기능 제공하는가 ?
- **신뢰성**
  - 실패없이 모든 기능/능력 제공? 필요시 이용가능? 오류 없는 기능 제공?
- **적합성**
  - 로컬및 외부 소프트웨어 표준 준수 ? 설계와 코딩 규칙 준수 ?
- **내구성**
  - 유지및 의도하지않은 부작용 발생시 부주의 없이 수정?
- **보수**
  - 허용 가능한 단시간에 변경 / 수정 / 변화 / 수정시 지원부서 필요 정보획득 ?
- **미학**
  - 명백한 특정 우아함 , 독특한 흐름 및 분명한 존재감 가지고 있는지 동의
- **지각**
  - 과거의 인식으로 부정적 / 긍정적 인식.

2.소프트웨어 품질

◆ 품질 **계수** : 소프트웨어 품질 에 **영향**을 미치는 요인

- 3 개의 중요한 소프트웨어 제품의 측면에 초점
  - **동작 특성 , 변화대처 능력 , 새로운 환경**



2.소프트웨어 품질

◆ 품질 **계수** : 소프트웨어 품질 에 영향을 미치는 요인

제품 측면	품질계수	내용
Product Operation	정확도	프로그램이 **고객 요구사항 충족**하여 미션 목표 완수 정도
	신뢰성	프로그램이 필요한 정밀도로 **의도된 기능 수행을 기대**하는 범위
	효율성	프로그램에 필요한 **컴퓨터 리소스및 코드의 양**
	무결성	권한없는 사람에게 조절되는 **소프트웨어나 데이터 액세스 정도**
	사용용이성	**배우고, 작동하고, 입력준비, 출력을 해석**하는데 필요한 노력
Product Revision	유지보수성	**오류를 찾아 수정**하는데 필요한 노력
	유연성	**동작 프로그램을 변경**하는데 필요한 노력
	테스트용이성	**의도된 기능을 보장**하는 프로그램을 테스트하는데 필요한 작업
Product Transition	이식성	**다른 하드웨어/소프트웨어시스템 환경**에서 프로그램 전송 필요 노력
	재사용가능성	프로그램 수행 기능의 **패키지**및 범위와 관련 **다른 애플리케이션에서 재사용** 범위
	상호운용성	**한 시스템에서 다른 시스템과 결합**하는데 요구되는 노력

2.소프트웨어 품질

◆ 품질 특성 : ISO 9126 품질 특성

- 기능성[Functionality]
  - 명시된 요구와 내재된 요구를 만족하는 기능을 제공하는 소프트웨어 제품 능력
- 신뢰성[Reliability]
  - 규정된 조건에 사용될 때 규정된 성능 수준을 유지할 수 있는 능력
  - 사용자가 오류를 방지할 수 있도록 하는 SW제품의 능력
- 사용성[Usability]
  - 능력사용자에 의해 쉽게 이해하고 학습되며 선호할 수 있게하는 SW제품의 능력
- 효율성[Efficiency]
  - 투입된 자원에 대하여 제공되는 성능의 정도
  - 요구되는 기능을 수행하기 위해 필요한 자원의 소요 정도
- 유지보수성[Maintainability]
  - 운영환경과 요구사항 및 기능적 사양에 따른 SW의 수정, 개선 등 변경될 수 있는 능력
- 이식성[Portability]
  - SW가 다른 HW,SW 등의 환경으로 옮겨갈 수 있는 능력
  - 다른 환경으로 이전되는 SW능력의 정도

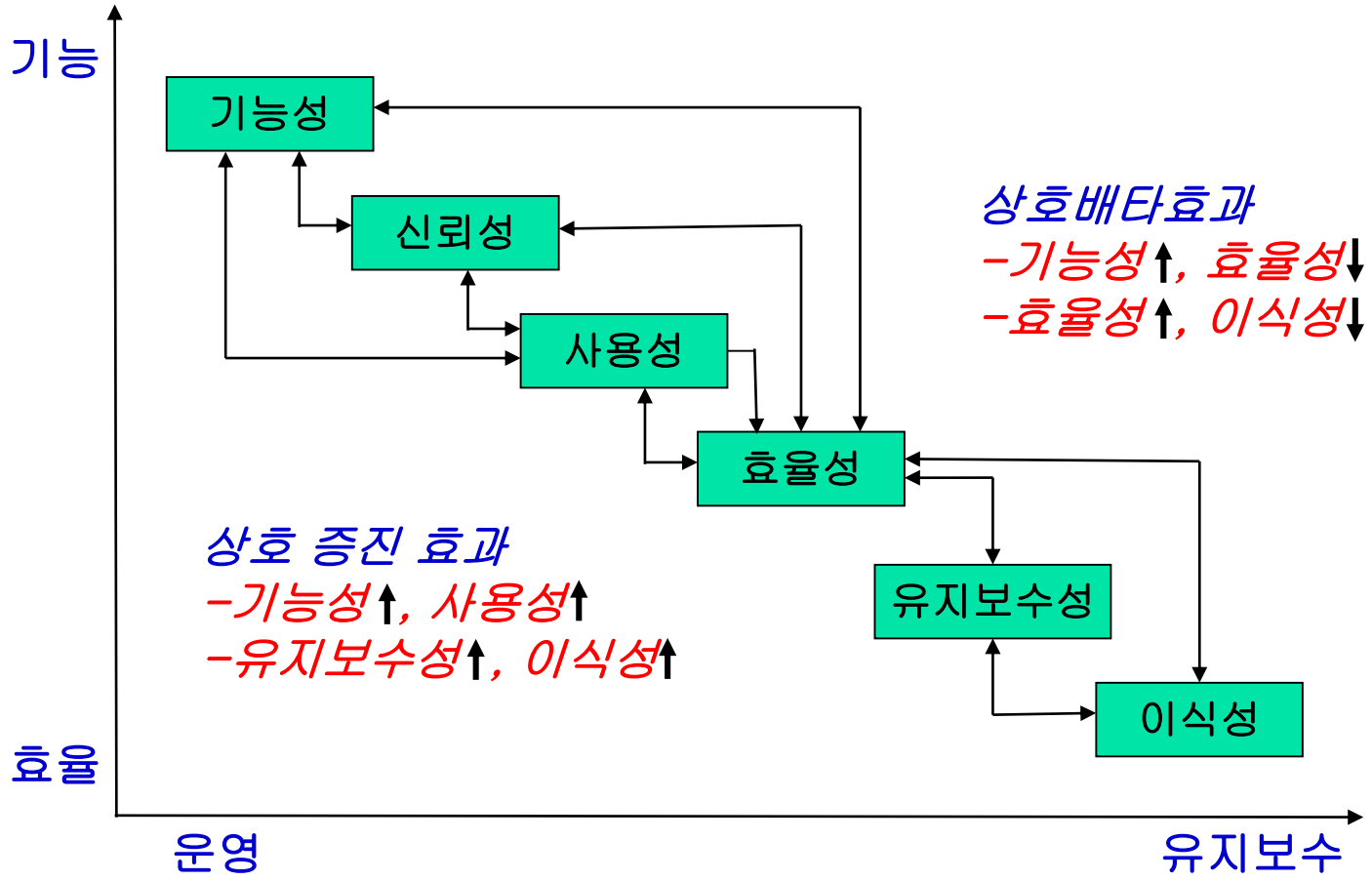
2.소프트웨어 품질

◆ 품질 **특성** : ISO 9126 품질모델(Quality Model)

사용자 관점에서 본 소프트웨어 품질 특성에 대한 표준

2.소프트웨어 품질

◆ 품질 **특성** : ISO 9126 품질특성 상호관계



3.소프트웨어 품질 딜레마

◆ 품질 딜레마

- 실패
  - 끔찍한 품질 소프트웨어 생산 -> 구입하는 사람이 없기 때문
- 사업 기회 손실
  - 절대 완벽한 소프트웨어 생산 : 무한의 시간 , 매우 큰 노력 , 거액 지출

◆ "만족스러운" 소프트웨어

- 최종 사용자 : 원하는 **고품질의 기능과 특색** 제공
- 소프트웨어 공급업체 입장 : 다른 애플리케이션 매우 만족하기 때문
  최종 사용자가 버그 간과 기대
- 회사에 영구적인 손상을 중 위험 -> 회사 폐쇄
  - 만족스러운 항공기 항공 소프트웨어 ?
- "만족스러운"소프트웨어 **품질 문제 해결할 수 있는 지름길** 이라고 신뢰
  - 신중하게 진행 하라
  - 작동하지만 제한된 애플리케이션 도메인 중 일부 적용 가능.

3.소프트웨어 품질 딜레마

◆ 품질/비용

- 논쟁
  - 원하는 소프트웨어 품질 수준 을 얻기 위한 많은 시간과 비용

◆ 품질 비용

- 품질을 추구하거나 품질 관련 활동
  - 품질의 부족으로 인한 후속 비용을 실행할 때 발생하는 모든 비용
- 품질 비용 분리 : 예방 , 평가 , 실패 관련 비용

1) 예방 비용

(1) 모든 품질관리 및 품질 보증 활동 을 계획 하고 조정 하는데 필요한 관리 활동 비용

(2) 전체 요구사항과 설계 모델을 개발하기 위한 추가 기술 활동 의 비용

(3) 테스트 계획 비용

(4) 이러한 활동과 관련된 모든 교육 비용.

3.소프트웨어 품질 딜레마

- 품질 비용 분리 : 예방 , 평가 , 실패 관련 비용

2) 평가 비용

(1) 소프트웨어 엔지니어링 작업 에 대한 기술 검토
(2) 데이터 수집 및 지표 평가 에 대한 비용
(3) 시험과 디버깅 에 대한 비용.

3) 실패 비용

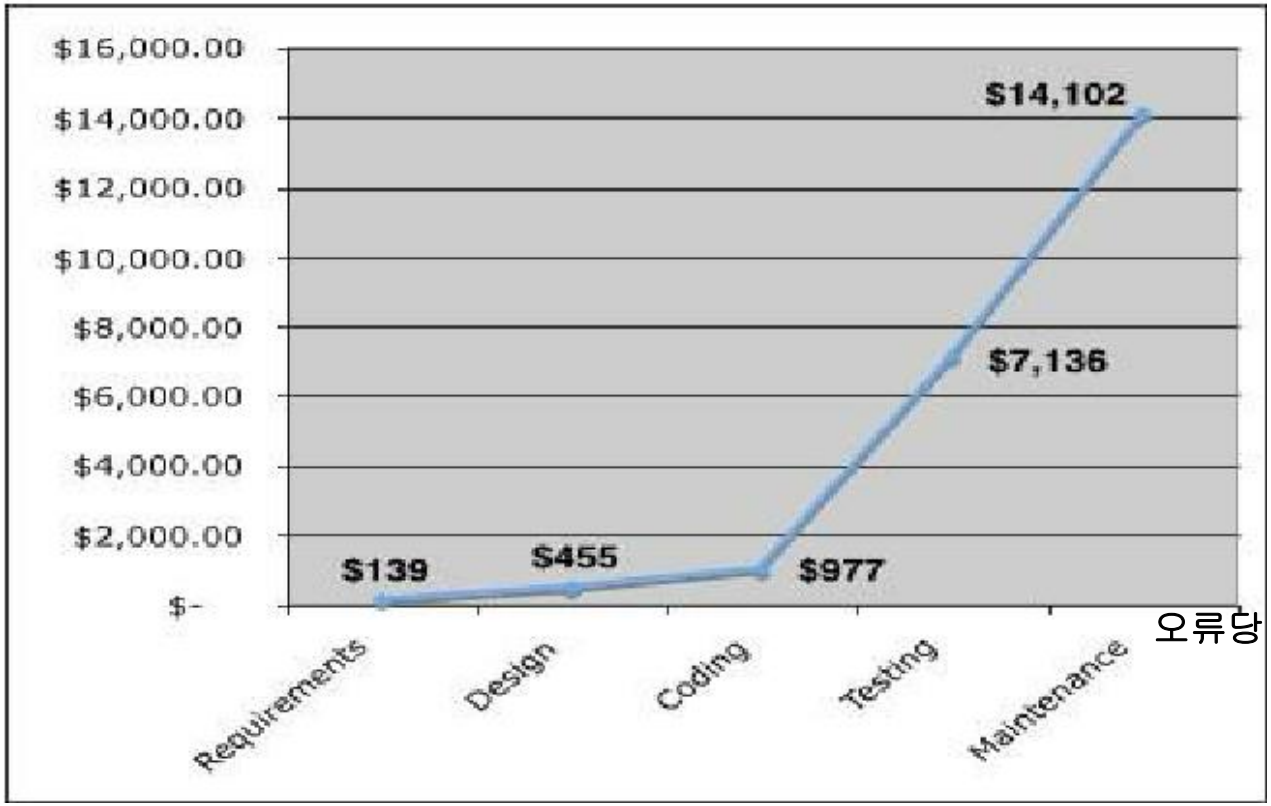
(1) 내부 실패 비용 : 출하전 오류 검출 비용
  - 오류를 정정하기 위해 재작업 (수리) 의 실행시 필요한 지용
  - 재작업 부주의로 인한 감소되어야 할 부작용을 생성할 때 발생 하는 비용
  - 실패 모드를 평가하기 위해 조직에 허용된 품질 측정 기준 수집 관련 비용

(2) 외부 실패 비용 : 고객 배송후 발견된 결함
  - 불만 해결, 제품의 반품및 교환, 고객 도움 라인 지원, 보증 업무와 관련된 인건비.

3.소프트웨어 품질 딜레마

◆ 오류 및 결함 수정의 상대적인 비용

- 예방에서 외부 고장 비용의 내부 결함 검출 로 갈 때
  오류나 결함을 복구 하는 상대적인 비용이 극적으로 증가



4.소프트웨어 품질 확립

- SW품질 : 우수한 프로젝트 관리 , 훌륭한 소프트웨어 엔지니어링 실행 결과
  - 1) 소프트웨어공학 방법
    - 고품질의 소프트웨어 구축 기대
      - 해결할 문제점 이해
      - 문제에 부합한 설계 작성 : 품질 치수와 설명 요인 적용
    - 문제와 포괄적 설계의 합리적이며 철저한 이해
      - 적절한 분석 및 설계 방법 채택.
  - 2) 프로젝트 관리 기술론
    - 프로젝트 계획에는 품질과 변경 관리를 위한 명백한 기술 포함
      - 매니저가 납기를 달성 여부를 검사하기 위해 추정 사용
      - 계획 종속성을 이해 하고 팀은 최소 노선 사용 유혹에 저항
      - 위기 기획이 실행 되어 문제가 혼란을 야기하지 않을 경우 긍정적 영향.

4.소프트웨어 품질 확립

- SW품질 : 우수한 프로젝트 관리 , 훌륭한 소프트웨어 엔지니어링 실행 결과
  - 3) 품질 관리 : 검증 활동
    - 품질목표를 충족하는지 확인 하기 위한 일련의 소프트웨어공학 조치포함
    - 모델들은 완전하고 일정한지를 확인 하기 위해 검토
    - 검사 시작 전에 오류를 발견하고 수정하기 위해 코드를 검사
      - 일련의 검사 단계 : 처리 로직 , 데이터 조작 , 인터페이스 통신 오류
      - 품질 미충족 : 측정과 피드백 -> 프로세스 조율.
  - 4) 품질 보증 : 시스템적인 활동
    - 훌륭한 소프트웨어공학 기술 , 합리적인 프로젝트 관리 , 품질관리 조치 를 지원하는 인프라 구축
    - 구성 : 품질관리 조치의 유효성을 평가하는 감사및 보고기능 세트
    - 목적 : 관리와 기술직원에게 제품 품질에 관한 필요한 정보를 통지 하고 , 그것에 의해 획득된 통찰력과 제품 품질을 얻을 수 있는 자신감 작용.

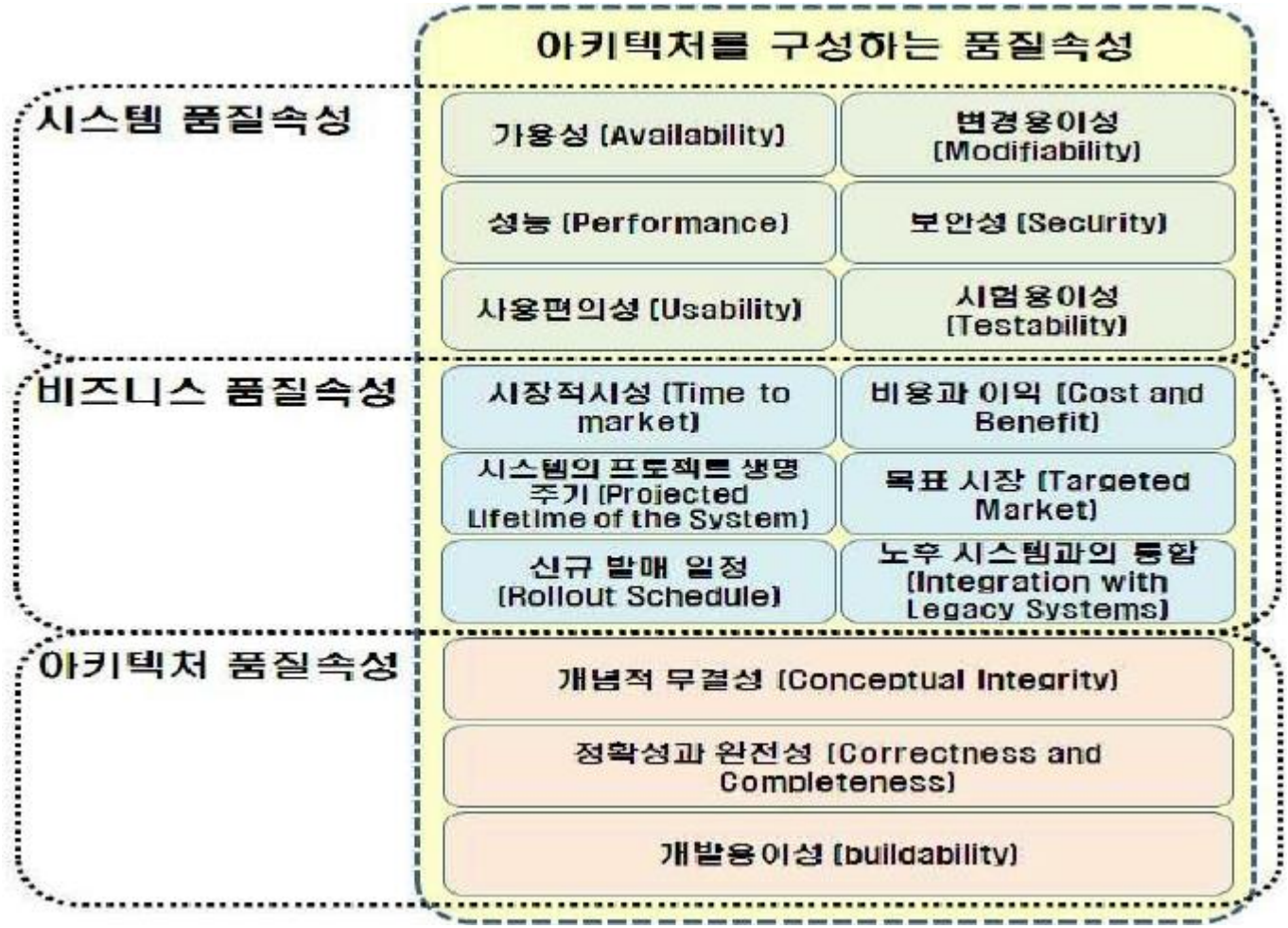
5.양질의 소프트웨어

◆ 프로세스의 품질

- 프로세스 개선을 위한 모델
  - 능력 성숙도 모델 (CMM : Capability Maturity Model)
    - 소프트웨어 개발 능력 측정 기준 과 소프트웨어 프로세스 평가 기준 을 제공함으로써 ,
    - 정보 및 전산 조직의 성숙수준을 평가 할 수 있는 모델
  - ISO 9000
    - 제품 생산 , 유통과정 전반 에 걸쳐 국제 규격을 제정한 품질보증제도
  - SPICE (Software Process Improvement and Capability dEtermination)
    - 소프트웨어 프로세스의 평가 를 위한 프레임워크.

6.품질 속성의 이해

◆ 품질 속성의 도식화



6.품질 속성의 이해

◆ 시스템 품질 속성

1) 가용성 (Availability)

- 시스템의 실패 (system failure) 와 이로부터 영향을 받는 것들과 연관
- 시스템이 더 이상 명시된 서비스를 제공하지 않는 경우 시스템 실패 가 발생
- 이러한 실패는 시스템의 사용자 (사람 혹은 타 시스템) 로부터 관측이 가능
- 시스템의 가용성이란 시스템이 필요할 때 운용될 수 있을 확률 로써 가용성이 99.9% 이면 시스템이 필요할 때 동작하지 않을 확률이 0.1%.

2) 변경용이성 (Modifiability)

- 변경에 대한 비용과 연관
- 특정 변경요구사항이 시스템에 반영 될 수 있도록 하는 소프트웨어의 능력
- 변경사항이 식별되면 새로운 구현이 설계 , 구현 , 테스트 , 배포 되어야 하므로 많은 시간과 비용이 소모.

3) 성능 (Performance)

- 성능은 주로 응답 시간 과 관련되며 , 컴포넌트 간에 얼마나 많은 상호작용 이 필요
- 컴포넌트마다 어떤 기능이 할당되는지, 공유 자원이 어떻게 사용
- 어떤 알고리즘이 구현.

6.품질 속성의 이해

◆ 시스템 품질 속성

4) 보안성 (Security)

- 보안성은 올바른 사용자에게 서비스가 제공되는 동안 승인되지 않은 사용에 대응 하는 시스템 능력의 정도

-보안을 파괴하려고 시도 하는 것을 공격 (Attack)

5) 시험용이성 (Testability)

- 소프트웨어가 용이하게 시험 될 수 있는 소프트웨어의 능력을 의미
- 결함을 찾아내기 위하여 시험이 얼마나 효과적 으로 수행
- 기대수준만큼의 범위를 시험하기 위해 소요되는 시간 이 얼마큼인지에 대한 것

6) 사용편의성 (Usability)

- 사용자가 원하는 작업을 수행하기 위해 시스템을 얼마나 쉽게 사용 할 수 있는가에 관한 것.

6.품질 속성의 이해

◆ 아키텍처 품질 속성

1) 개념적 무결성 (Conceptual integrity)

- 개념적 무결성은 흔히 일관성 이라고도 함
- 모든 레벨의 시스템 설계를 통합하는데 근간이 되는 개념
- 아키텍처는 유사한 방법으로 유사한 일들을 수행 해야 하고, 일괄된 방식으로 적용 될 수 있어야 함.

2) 정확성과 완전성 (Correctness and Completeness)

- 두가지 품질속성은 아키텍처가 시스템의 요구사항과 런타임 자원에 대한 제약사항 모두를 만족 해야 한다는 가장 중요한 품질속성.

3) 개발용이성 (Buildability)

- 구축 가능성은 개발용이성 을 말하는 것으로 개발할 능력이 있는 팀이 적절한 시기에 시스템을 완성
- 개발이 진행되면서는 특정 변경이 가능 하도록 하는 특성.

정리 및 Homework

1) 품질 계수
2) 품질 치수
3) 품질 특성
4) 품질 비용
5) 품질 관리와 보증

Project

1장. 프로젝트 개요
- 1.1 프로젝트 제목
- 1.2 선정 이유
- 1.3 팀 운영 방법

2장 시스템 정의
- 2.1 시스템 간략한 설명
- 2.2 유사 사례 간략한 설명

3장 프로세스 모델
- 3.1 규범적인 프로세스 모델 선정 및 이유
- 3.2 특수한 프로세스 모델 선정 및 이유

4장. 실무 가이드 원칙
- 4.1 각 프레임워크 원칙에서 중요한 3 개 정의
- 4.2 프로젝트 계획 보고서

5장. 요구사항 획득
- 5.1 기능 요구사항과 비기능 요구사항 정의
- 5.2 표준 양식을 사용한 시스템 요구사항 명세 3개 작성
- 5.3 정형적인 형식에 따른 유스케이스 작성

6장. 시스템 설계
- 6.1 설계 개념의 중요한 개념을 적용
- 6.2 설계 모델에 따른 요소별 설계

7장. 아키텍처 개념
- 7.1 아키텍처 스타일 선정및 이유
- 7.2 아키텍처 설계 프로세스 정의 및 설계

8장. 품질
- 8.1 시스템 품질 속성 정의
  (개인당 2개씩정의하여 팀당 3개씩결정)
- 8.2 비즈니스 품질 속성 정의
  (개인당 2개씩정의하여 팀당 3개씩결정)
- 8.3 아키텍처 품질 속성 정의
  (개인당 2개씩정의하여 팀당 2개씩결정)