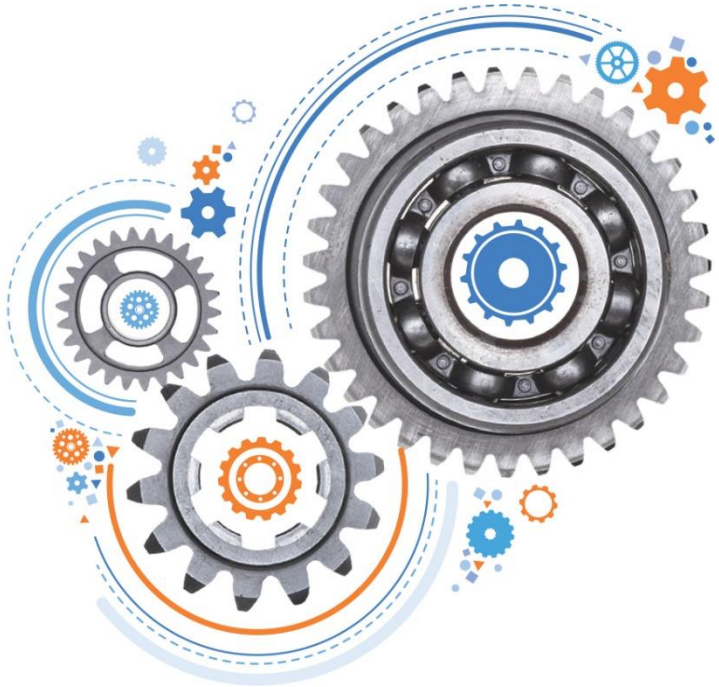

시스템 분석과 설계 개정판

효과적인 비즈니스 정보시스템 개발

Chapter 16
소프트웨어 품질관리

목차

학습목표

- 소프트웨어 품질관리의 개념 및 품질목표에 대해 학습한다.
- 소프트웨어 품질관리를 위한 절차를 알아본다.
- 경험적 품질관리 기준을 통해 품질관리를 실제적인 개발 프로젝트에 적용할 수 있도록 학습한다.

1.1 소프트웨어 품질관리의 개념

소프트웨어의 전반적인 문제점

- 응답성(Responsiveness) : 사용자의 요구에 부응하지 못하는 소프트웨어
- 신뢰성(Reliability) : 잦은 오류
- 비용(Cost) : 예상보다 과도한 비용
- 변동성(Modifiability) : 소프트웨어의 수정은 복잡하고, 많은 비용을 수반
- 적시성(Timeliness) : 소프트웨어의 개발 지연
- 변환성(Transportability) : 다른 목적에 맞게 변환하기 쉽지 않음
- 효율성(Efficiency) : 자원을 효율적으로 사용하지 못함

소프트웨어 품질 정의

- 소프트웨어의 품질이란 요구사항 또는 제품사양에 대한 적합성을 의미한다.
- 소프트웨어의 품질이란 사용 중 발생하는 실패의 정도로 정의한다.
- 소프트웨어의 품질이란 하나의 제품 혹은 서비스가 목표하고 있는 사용 목적을 충족시켜 줄 수 있는 능력에 대한 전체적인 특징 및 특성을 말한다.

1.1 소프트웨어 품질관리의 개념

- **소프트웨어 품질이란 다음의 요구사항들을 충족시켜주며, 결함이 없는 소프트웨어를 보장하는 것을 의미함**
- **요구사항**
  - 제공하는 기능
  - 성능표준
  - 표준 대비 성능측정 정의

■ 결함이란 요구사항을 충족시키지 못하는 실패 요소들을 의미함

표 16-1 결함의 주요 요소

주요 요소	내용
환경	소프트웨어의 실행 환경이 열악하여 소프트웨어가 제 성능을 발휘하지 못하는 경우를 예로 들 수 있다. 네트워크 환경의 불안정, 서버의 용량이나 성능 미비, 필요한 장비(Hardware)의 미비 등이 이에 속한다.
역할분담	회사의 조직이나 부서는 업무의 원만한 처리를 위해 필수적인 요소라 할 수 있다. 마찬가지로 소프트웨어의 개발과 운영에 따른 적절한 조직과 부서를 두어 역할을 분담해야 한다.
방법과 절차 등의 도구	표준화된 방법과 절차 등이 마련되어 운영된다면 소프트웨어의 안정적인 운영이 가능할 것이다. 반면에 그러한 방법이나 절차없이 임의로 운영된다면 혼란과 과오를 범할 확률이 매우 높다.
시간과 인력 등의 자원	소프트웨어의 개발과 운영에는 두 개의 큰 요소(자원)가 필요한데 시간과 인력이 바로 그것이다. 소프트웨어의 개발 및 운영비용 역시 시간과 인력을 요소로 한 계산 값(MM : Man Month)이 활용된다.
훈련	소프트웨어 운영을 위해서는 운영자에 대한 충분한 훈련이 뒤따라야 한다. 훈련되지 못한 운영자에 의해 얼마나 많은 실수와 과오가 저질러지는지를 감안한다면 훈련의 중요성을 간과해서는 안 될 것이다.

1.1 소프트웨어 품질관리의 개념

■ 소프트웨어의 품질

- 효과와 효율의 측면에서 측정 가능
  - 효과 : 요구사항과의 일치 정도
  - 효율 : 품질비용과 자원 사용량 등에 의해 측정

「Quality cannot be achieved unless it can be measured, and it cannot be measured unless it can first be defined.」

1.1 소프트웨어 품질관리의 개념

소프트웨어 품질 요구사항의 특성

- 다차원적
- 서로 다른 요구사항 간의 갈등관계
- 비즈니스 요구사항과의 상충
- 표현하기 어려움
- 이득을 쉽게 측정하기

■ 소프트웨어의 품질목표

표 16-2 **소프트웨어 품질목표** [02]

품질목표	정의
효율성(Efficiency)	최소한의 컴퓨터 시간과 기억장소를 소요하여 요구된 기능을 수행하는 시스템 능력
융통성(Flexibility)	새로운 요구사항에 접하여 쉽게 수정될 수 있는 시스템 능력
무결성(Integrity)	시스템 소프트웨어나 데이터의 독단적인 접근 및 수정을 제어할 수 있는 시스템 능력
상호운용성(Interoperability)	다른 시스템과 정보를 교환할 수 있는 시스템 능력
유지보수성(Maintainability)	에러가 발견되었을 때 쉽게 정정할 수 있는 시스템 능력
이식성(Portability)	하나 이상의 하드웨어 환경에서 운용되기 위해 쉽게 수정될 수 있는 시스템 능력
신뢰성(Reliability)	정확하고 일관된 결과로 요구된 기능을 수행하는 시스템 능력
정확성(Correctness)	사용자의 요구 기능을 충족시키는 정도
재사용성(Reusability)	시스템의 일부나 전체를 여러 가지 응용 부분에서 사용할 수 있는 능력
테스트용이성(Testability)	쉽고 완전하게 테스트할 수 있는 시스템 능력
사용용이성(Usability)	쉽게 배우고 사용할 수 있는 시스템 능력

1.3 소프트웨어 품질보증을 위한 접근방법

■ 소프트웨어 품질공학의 구조

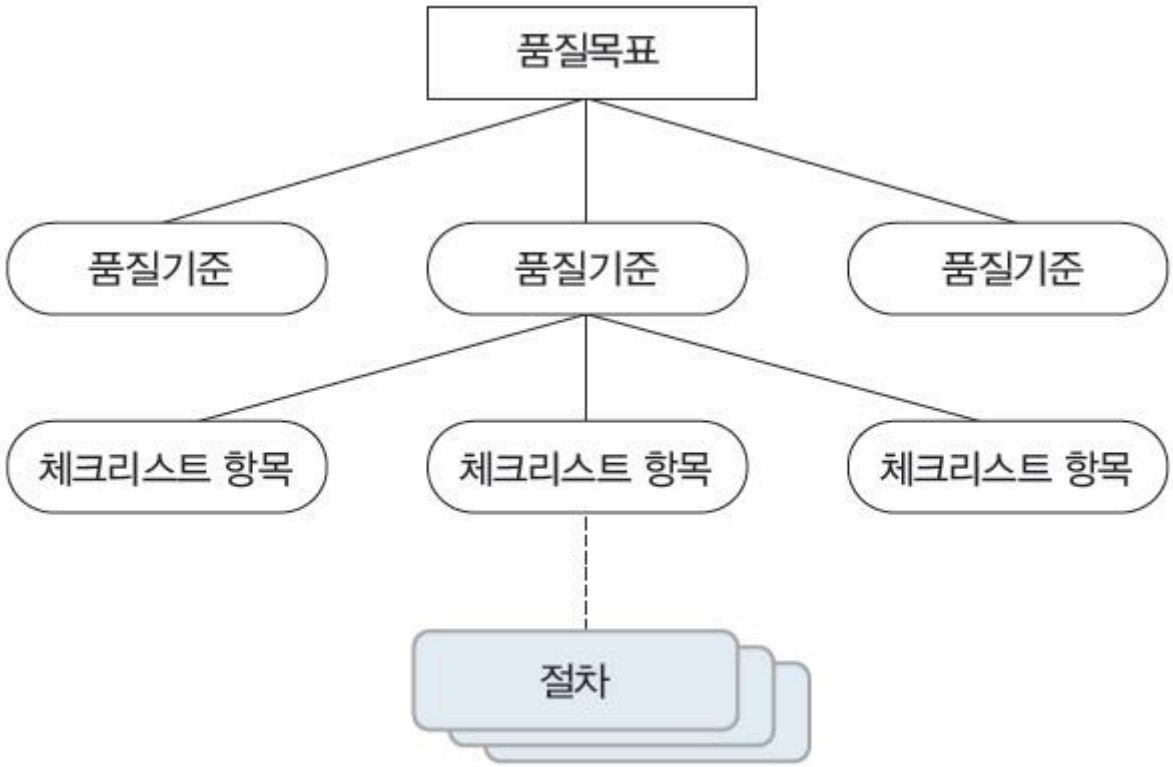

그림 16-1 소프트웨어 품질공학의 프레임워크 [03]

■ 품질목표와 소프트웨어 개발 단계와의 관계

표 16-3 품질목표와 생명주기 단계와의 관계

생명주기 단계 / 품질목표	개발			평가	응용 및 유지보수			투자 효율성
	요구 분석	설계	구현	시스템 시험	운용	수정	변형	
정확성	△	△	△	×	×	×		높음
신뢰성	△	△	△	×	×	×		높음
효율성	△	△	△		×			낮음
무결성	△	△	△		×	×		낮음
사용용이성	△	△		×		×		보통
유지보수성		△	△			×	×	높음
시험용이성			△	×	×	×	×	높음
유연성		△	△				×	보통
이식성		△	△				×	보통
재사용성		△	△				×	보통
상호운용성	△	△		×			×	낮음

△ : 품질목표달성도 측정 시기, × : 저품질의 영향이 나타나는 시기

ISO 품질목표 및 품질기준

표 16-5 ISO 품질목표 및 품질기준 [05]

품질목표	정의	품질기준
기능성 (Functionality)	명확한 이용자의 요구를 만족하는 기능의 존재와 특성에 관한 속성	정확성, 안전성, 호환성, 접속성 등
신뢰성 (Reliability)	정해진 기간과 조건 하에서 그 성능수준을 유지하기 위한 능력과 관계있는 속성	무결함성, 오차허용성, 가용성 등
사용성 (Usability)	소프트웨어를 사용하는 데 필요한 노력 및 특정(혹은 불특정) 사용자의 사용평가에 관한 속성	이해성, 조작성, 대화성 등
효율성 (Efficiency)	정해진 조건 아래에서 소프트웨어 제품의 일정한 성능과 자원 소요량의 관계에 관한 속성	시간 경제성, 자원 경제성 등
유지보수성 (Maintainability)	소프트웨어 변경 시 필요한 노력과 관계되는 속성	수정 용이성, 확장성, 테스트 용이성 등
이식성 (Portability)	소프트웨어를 다른 환경으로 이식할 경우에 관계되는 속성	HW독립성, SW독립성, 도입용이성, 재사용성 등

2.1 소프트웨어 품질관리 계획

■ 품질관리 단계

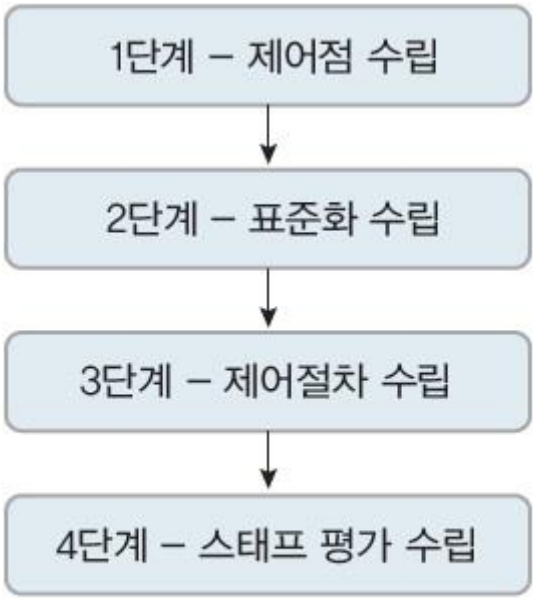

그림 16-2 품질관리의 4단계

2.1 소프트웨어 품질관리 계획

■ 품질보증 계획서

- 계획의 목적과 범위
- 계획서에서 참조된 문서들
- 조직 구조, 수행될 작업, 생산물의 품질과 관련된 특별한 임무
- 준비해야 할 문서와 그 문서의 적합성 검토
- 사용될 표준, 규범과 관례
- 검열과 검사
- 형상(Configuration) 관리 계획
- 소프트웨어 문제점을 기록하고 추적하여 해결하기 위한 규범과 수행절차
- 품질보증 활동을 지원하기 위한 특별한 도구와 기법
- 특정 소프트웨어 버전을 유지하고 저장하기 위한 방법과 설비
- 물리적 장치로부터 컴퓨터 프로그램을 보호하기 위한 방법과 설비
- 벤더(Vendor)가 제공하거나 하청업체가 개발한 소프트웨어의 질을 확인하는 설비
- 품질보증 기록을 수집, 유지 그리고 보존하기 위한 방법과 설비

2.1 소프트웨어 품질관리 계획

품질관리팀의 조직 관리

- 프로젝트 조직 : 의사결정이 빠르고 인터페이스 최소화
  프로젝트 크기가 작을 때만 적용 가능
- 기능적 조직 : 전문성을 잘 살릴 수 있지만, 인터페이스가 너무 많아짐
- 매트릭스 조직 : 프로젝트 조직과 기능적 조직을 혼합
  책임이 분산되는 경향이 있음

2.2 소프트웨어 품질보증 단계별 도구

■ 품질보증 단계

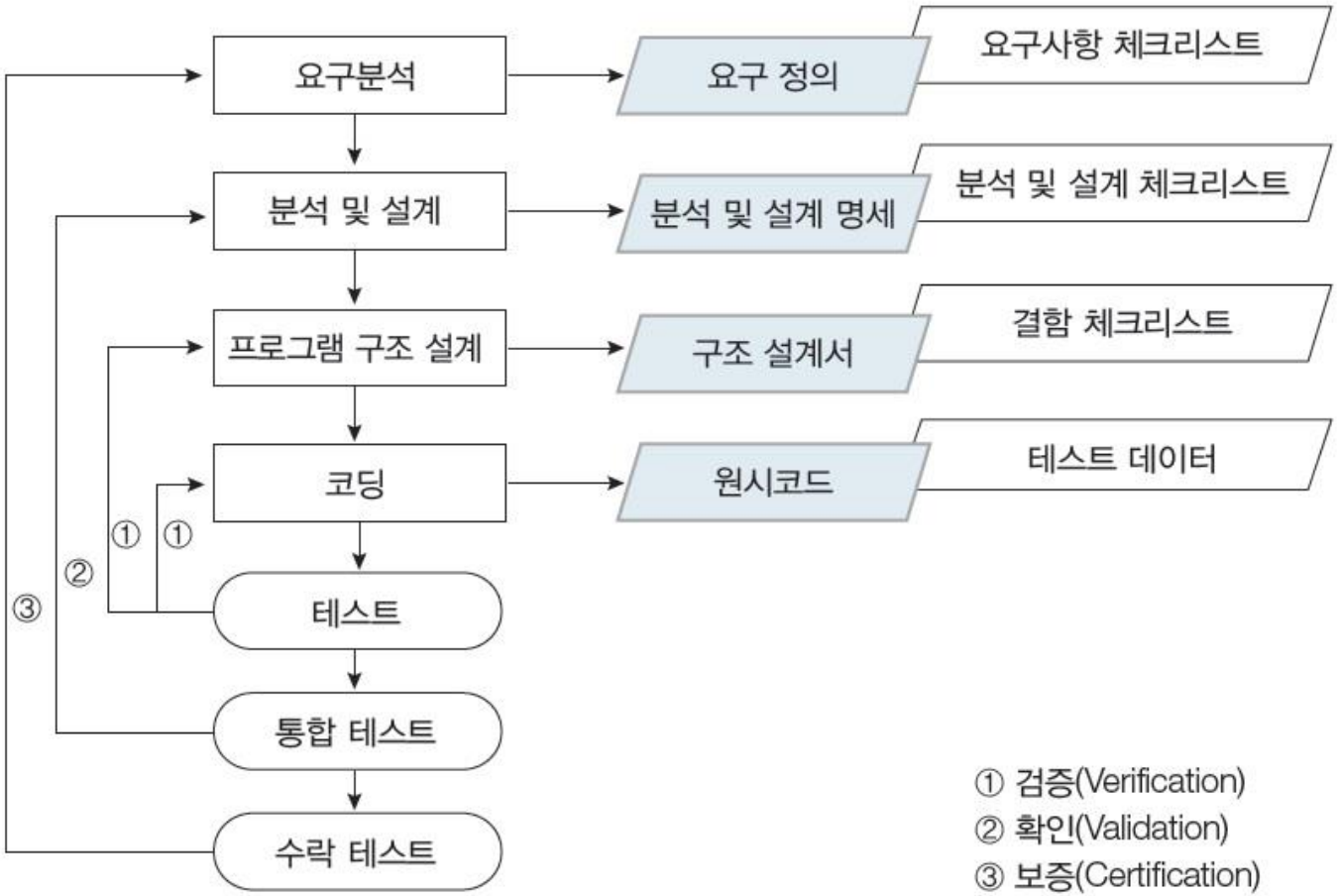

그림 16-3 품질보증 단계 [06]

2.2 소프트웨어 품질보증 단계별 도구

요구분석 단계의 품질 보증

- 사용자의 요구사항 명세서를 기준으로 평가

설계 단계의 품질보증

- 자료구조 설계에 중점을 두어 평가

코드의 품질보증

- 블랙박스 테스트 방법
- 화이트박스 테스트 방법

2.2 소프트웨어 품질보증 단계별 도구

검증과 확인

- 검증 : 소프트웨어가 고객의 요구사항을 만족시키는가의 여부를 밝히는 활동
- 확인 : 소프트웨어가 지정된 기능에 대해 정확하게 수행되는가의 여부를 확인

검토과 검열

- 검토 : 구조적 검토회의에서 개발 담당자와 사용자가 각 단계의 산출물 검토
- 검열 : 조정자에 의해 주관되며 검열팀에는 조정자, 설계자, 프로그래머 및 테스트 전문가가 참여

→ 검토와 검열은 인간에 의해 진행된다는 특징이 있음

2.2 소프트웨어 품질보증 단계별 도구

■ 검열의 진행단계

그림 16–4 검열의 진행 단계

3.2 경험적 품질관리를 위한 기준

개발자 측면

- 사용자의 입장을 고려한 사용편의성, 기능성, 신뢰성 등에 역점을 두어야 함
- 다양한 환경에 적응할 수 있도록 이식성을 중시해야 함
- 유지보수가 용이한 소프트웨어를 개발하는 데 우선순위를 두어야 함
- 문서화에 철저해야 함
- 변경사항 관리는 관리자의 통제 하에 이루어지는 것이 바람직

3.2 경험적 품질관리를 위한 기준

사용자 측면

- 개발의 각 단계별로 산출되는 결과물에 대한 검증과 확인 작업을 소홀히 하지 않아야 함
- 사용자 운영규정을 꼼꼼히 살핌
- 참조 매뉴얼을 요구

3.2 경험적 품질관리를 위한 기준

신뢰성 측정의 척도

MTBF = MTTF + MTTR

- MTBF : 실패 평균시간(Mean Time Between Failure)
- MTTF : 평균 실패시간(Mean Time To Failure)
- MTTR : 평균 보수시간(Mean Time To Repair)

이용가능성

이용가능성(Availability) = MTTF / (MTTF+MTTR) * 100%

Thank You