

1 유지보수의 필요성

01 유지보수의 필요성

1 유지보수가 필요한 이유

 소프트웨어를 끊임없이 개선해 나가는 과정이 필요

- 대체로 프로젝트 수행기간보다 훨씬 더 오랜 기간 동안 인력과 비용이 투입
- 실제 비즈니스 경쟁력을 갖추기 위한 노력은 유지보수 과정에서 발생

 유지보수는 쉽지 않음

- 다른 사람이 작성한 프로그램을 이해하기 쉽지 않음
- 시간이 지날수록 문서 산출물이 현실과 맞지 않음
- 선호되는 직종이 아님

01 유지보수의 필요성

1 유지보수가 필요한 이유



소프트웨어 품질문제

- 사용자의 만족에 미흡
- 개발과정에 사용자의 참여 부족
- 소프트웨어의 신뢰성 부족
- 추가개발 과다
- 재사용을 고려하지 않은 개발

01 유지보수의 필요성

1 유지보수가 필요한 이유



조직의 내외부적인 문제

- 경쟁력을 갖추기 위한 노력
- 법/제도적 변화
- 조직의 급격한 성장
- 분사 혹은 조직구조의 변화
- 사업의 다각화

01 유지보수의 필요성

2 유지보수 활동의 분류

분류	내용
단순운영	시스템 가동 모니터링 데이터 백업, 보관, 삭제 정기, 비정기 배치작업
소스코드 변경	단순 기능개선을 위한 소스코드 수정 기능의 변경 없는 소프트웨어 성능향상 Bug 제거 및 오류 수정
기능개선	새로운 기능의 추가에 의한 기능개선
장애대응	장애 예방활동, 장애발생시 긴급대응, 장애해결

01 유지보수의 필요성

3 유지보수 아웃소싱



전문 아웃소싱 업체에게 유지보수를 의뢰

- 기업은 전문적인 핵심 비즈니스에 집중
- 계약에 의해 서비스 수준을 지정하고 품질을 보장
- SLA(Service Level Agreements)
: 적정 서비스 수준을 계약으로 명시

01 유지보수의 필요성

4 SLA

- 🔍 고객사에게 제공하는 정보시스템의 관리수준을 정량적으로 측정하여 성과에 대한 평가, 개선 및 대가산정의 기준으로 삼는 방식
- 🔍 고객에 대한 신뢰도와 만족도를 향상시키는 효과
- 🔍 서비스 품질의 계량적 관리는 객관적 증빙이 가능
- 🔍 의사소통도 명확해지는 효과

01 유지보수의 필요성

4 SLA

유지보수 서비스 분류

서비스	내용
고객지원 및 응대	<ul style="list-style-type: none">고객지원 콜센터 운영
업무 지속성	<ul style="list-style-type: none">장애관리, 변경관리, 프로그램 개선
시스템품질	<ul style="list-style-type: none">온라인 응답속도(가용성)배치작업 처리시간(적시성)
기술환경가동	<ul style="list-style-type: none">네트워크 지원서버 운용PC관리 기술지원
보안	<ul style="list-style-type: none">기술적 보안, 물리적 보안, 관리적 보안

01 유지보수의 필요성

4 SLA

- 유지보수 서비스 수준
 - 계약된 서비스 수준 이상을 달성 : 보너스(Earn-back)
 - 서비스 수준에 미달 : 패널티(Penalty)

서비스	내용
목표 초과	<ul style="list-style-type: none">서비스 이용료 = 기본 서비스료 + earn back 발생
목표 달성	<ul style="list-style-type: none">서비스 이용료 = 기본 서비스료
목표 미달	<ul style="list-style-type: none">서비스 이용료 = 기본 서비스료 - penalty

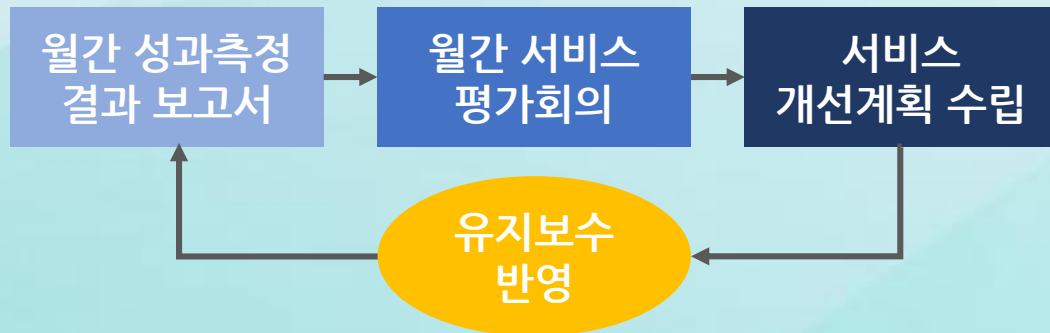
01 유지보수의 필요성

4 SLA



월간 서비스 평가회의

- 서비스에 대한 평가를 위한 고객과의 의사소통 채널
- 성과를 검증하고 개선사항을 도출



2 재공학

02 재공학

1 재공학의 필요성



소프트웨어 재사용의 필요성

- 이미 개발된 소프트웨어를 반복 사용하여 생산성을 향상시키는 방법으로 재사용 가능한 부분을 찾아 재사용 가능한 코드로 발전시키는 것



소프트웨어 재사용의 장점

- 개발 기간과 비용의 단축
- 프로젝트 실패의 위험 감소
- 개발 지식의 공유 가능
- 소프트웨어 개발의 생산성 향상

02 재공학

1 재공학의 필요성



소프트웨어 재공학의 정의

- 기존의 소프트웨어를 파기하는 대신에 사용자 요구사항의 변경 또는 수정된 환경으로 기존 소프트웨어를 수정, 보완하여 재구축하는 기술



소프트웨어 재공학의 출현 배경

- 소프트웨어의 위기를 개발의 생산성이 아닌 유지보수의 생산성으로 해결하기 위해 출현
- 재사용하는 과정에서 기존 소프트웨어의 다양한 문제점을 발견하고 이를 수정할 수 있음

02 재공학

1 재공학의 필요성





소프트웨어 재공학의 목적

- 유지보수성, 생산성, 품질의 향상
- 소프트웨어 수명 연장, 비용 감소, 기술 향상
- 복잡한 소프트웨어 학습 및 분석 가능
- 누락되거나 잃어버린 부분 복구 가능
- 잘못된 부분 수정 가능
- 재사용 기술의 습득 가능

02 재공학

2 재공학의 대상

-  재공학의 대상은
“비즈니스 프로세스(Business Process)”
-  비즈니스 프로세스의 구조적인 재설계를 통해
혁신적으로 일하는 방식을 바꾸어야 함을 말함

02 재공학

2 재공학의 대상



마이클 해머의 말

- “고객에게 가치를 제공하지 못하는 업무는 제거되어야 하며, 자동화를 통해 가속화되어서는 안된다”
- 기업은 고객의 가치를 극대화하기 위해 프로세스를 재편하고 상품과 서비스의 제공을 위한 자원을 절약해야 한다”

➤ BPR(Business Process Re-engineering)의 탄생

02 재공학

3 BPR의 정의

🔍 획기적인 결과를 성취하기 위하여
비즈니스 프로세스를 근본적으로 변경하려는
노력과 구현[포춘(Fortune)]

[재공학 사이클]



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사

02 재공학

4 재공학 사이클

1 목록분석

- 소프트웨어 목록을 정리하고 중요도에 따라 정렬, 후보 소프트웨어의 우선순위를 선정

2 문서 재구성

- 소프트웨어에 대한 문서의 최신화

3 역공학

- 설계형상을 추출



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사

02 재공학

4 재공학 사이클

- 4 코드의 재구성
- 소스코드 분석 및 재구성하고,
문제가 있는 프로세스를 변경하여 반영

- 5 데이터 재구성
- 데이터구조를
분석하여 개선

- 6 순공학
- 재공학 엔진으로
소스코드 생성 및 가동



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사

3 역공학

03 역공학

1 역공학의 정의 및 목적



정의

- 완제품으로부터 설계를 해독해 내는 작업
- 하드웨어의 역공학 → 복제
- 소프트웨어에서의 역공학 → 유지보수 및 기능향상



목적

- 스파게티 소스코드, 암호화된 소스코드, 중복된 소스코드 정리
- 소프트웨어 유지보수 비용을 줄임
- 노후 시스템(Legacy system)을 새롭게 변혁(Renovation)

03 역공학

2 역공학의 의미와 종류



프로그램 소스코드로부터
설계정보를 추출해내는 것을 의미



추상화된 모델을 추출

- 데이터 구조
- 프로세싱(일명 처리로직, 그리고 기능)
- 유저 인터페이스

03 역공학

3 데이터 구조 역공학



내부 데이터 구조

- 프로그램의 변수를 추출하는 것
- 변수를 대상으로 객체 클래스를 정의



데이터베이스 구조

- 관계형 데이터베이스의 구조를 재정의
- DBMS제품별 자동화 정도가 다름

03 역공학

4 프로세싱 역공학



기능적인 추상화

- 기능단위 블록 다이어그램
(모듈에 대한 설명이 추출)
- 모듈 간의 관계도로 표시



처리로직의 추상화

- 모듈의 내부를 추상화
- 데이터 준비, 처리, 외부로 인터페이스 등의
패턴을 가시적으로 보여줌

03 역공학

5 사용자 인터페이스 역공학

- 🔍 사용자 인터페이스(UI)에서 나타나고 있는 특징을 새로운 표준으로 분류 및 재조합
- 🔍 새로운 원칙에 따라 기존의 UI를 새롭게 재구조화
- 🔍 UI를 재구조화 하는 이유
 - UI는 유행과 신기술 적용에 민감
- 🔍 UI의 가독성을 높이고 성능을 향상시킬 수 있는 방향으로 재설계