소프트웨어공학 활용

구현 및 유지보수 도구

한국기술교육대학교
온라인평생교육원

학습내용

- 프로그래밍 및 프로토타입 도구
- 품질 보증 및 유지보수 도구

학습목표

- 프로그래밍 도구, 프로토타입 도구를 이해하고 설명할 수 있다.
- 품질 보증 및 유지보수 도구에 대하여 이해하고 설명할 수 있다.

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

1 프로그래밍 도구(소프트웨어 도구)

1 개요

프로그래밍 도구란?

소프트웨어 개발자가 다른 프로그램과 응용 프로그램을 생성·오류 수정·유지보수 하는데 사용하는 프로그램이나 응용 프로그램

2 프로그래밍 도구 선택 기준

1. 개발 언어(c, java, html, java script, object c 등)에 따라 선택
2. OS(window, Unix, Linux, mac 등) 종류에 따라 선택
3. 언어마다 사용되는 목적이 존재하고 그에 따라 사용해야 할 기능이 달라짐

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

1 프로그래밍 도구(소프트웨어 도구)

3 Unix 시스템의 기본 개발 도구 vi, vim

1. 가장 기본적인 개발 도구
2. vim은 vi의 기능이 확장된 프로그래밍 도구
3. Unix 시스템의 기본으로 설치되어 있음
   - ➡ MAC, Linux에서 동작함
4. ssh를 통한 원격 접속을 통해 개발하기 쉬움
   - ➡ GUI 요소가 적음

4 C 언어 개발을 위한 Microsoft Visual Studio

1. c 언어 개발을 위한 가장 대중적인 프로그램
2. 통합 개발 도구로써 다양한 언어 개발 지원
   - ➡ c, c++, c#, python, 비쥬얼 베이직 등
3. 최근에는 반응형 웹페이지 개발에 도움이 되는 기능 제공

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

1 프로그래밍 도구(소프트웨어 도구)

5 Java, 안드로이드 개발을 위한 eclipse

1. java 언어 개발을 위한 가장 대중적인 프로그램
2. 서버 쪽 개발에 자주 사용됨
   - ➡ git, svn과 같은 형상 관리 도구와 연동이 쉬움
3. 데이터베이스와 연동이 쉬움
4. 이클립스 상에서 안드로이드 개발 SDK 추가 설치를 통한 확장성이 좋음

6 Front-end 개발용 WebStrom

1. Front-end(Client) 개발에 강점이 있음
2. 서버 쪽 개발에도 여러 가지 편리한 기능이 있음
3. Git을 이용한 형상 관리 및 CSS, HTML 자동 완성기능 지원
4. 개발 도구 내에 Grunt가 내장되어 있음
   - ➡ 코드 수정 시 자동 컴파일, 백업, 테스팅까지 진행 가능함

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

- Anjuta
- Code::Blocks
- eclipse
- Microsoft Visual Studio(상용) & Microsoft Visual Studio Express
- NetBeans

1 Anjuta

개발사	• GNOME의 Anjuta project
설명	• GNOME 프로젝트를 위한 IDE • 프로젝트 관리, 위자드를 통한 앱 생성 • 디버깅 지원, 형상 관리 연동, GUI designer 포함 • Ubuntu, openSUSE, Fedora, Mandriva Linux 배포 패키지에 포함됨

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

1 Anjuta

플랫폼	• Linux
지원 언어	• C, C++, Java, JavaScript, Python

2 Code::Blocks

개발사	• Code::Blocks
설명	• GUI 툴킷인 wxWidgets으로 구현된 개발 환경 지원 도구 • GCC, Visual C++을 포함해 다양한 컴파일러들과 연동 • plugin 구조로 여러 가지 언어 개발 지원

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

2 Code::Blocks

플랫폼	• Linux
지원 언어	• C • C++ • Fortran

3 eclipse

개발사	• Eclipse Foundation
설명	• Standard IDE 외에 Java EE, Modeling, DSL, Automotive 등 전문분야 개발자를 위한 버전을 제공 • plugin을 통해 방대한 기능을 제공 • IDE를 넘어선 하나의 개발 플랫폼

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

3 eclipse

플랫폼	• 크로스 플랫폼
지원 언어	• C, C++, Java, JSP, PHP, Fortran, python 등

4 Microsoft Visual Studio(상용) & Microsoft Visual Studio Express

개발사	• Microsoft
설명	• Editor, 디버깅, 소스 버전 관리 시스템 연동 • Team Foundation Server와 연동 • Express는 Shareware 배포 버전으로 제한된 기능을 지원하지만 빌드 기능을 제공 • Visual Studio Express 버전별로 언어 지원 상품이 따로 존재

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

4 Microsoft Visual Studio(상용) & Microsoft Visual Studio Express

플랫폼	• Window
지원 언어	• C, C++ • Java • JSP • PHP • Fortran • python 등

5 NetBeans

개발사	• Oracle Corporation
설명	• NetBean IDe는 JavaFx를 포함하는 모든 Java SE 애플리케이션 지원 • Java Me, web, EJB, mobile 애플리케이션 개발 지원 • Maven 빌드 지원, 형상 관리 연동, 디버깅 내장

프로그래밍 및 프로토타입 도구

1 프로그래밍 도구

2 통합 개발 도구

5 NetBeans

플랫폼	• 크로스 플랫폼
지원 언어	• C, C++, Java, JSP, PHP, HTML5, Java Script

2 프로토 타입 도구

1 프로토타입(Prototype)

1 개념

프로토 타입이란?

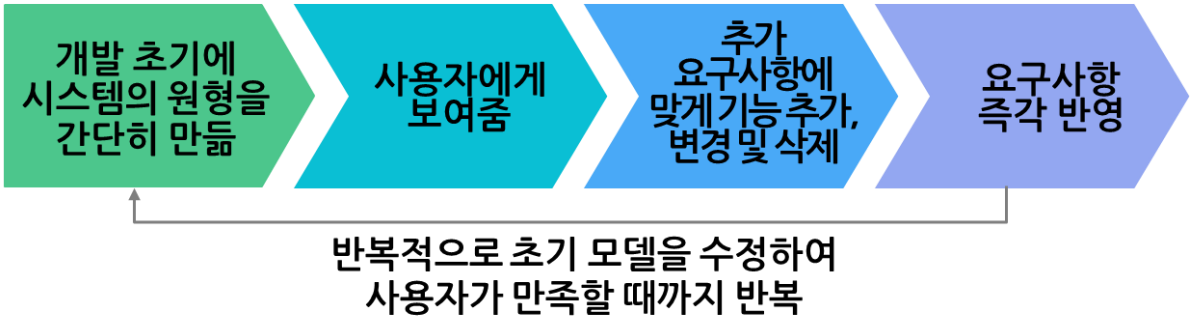
- 원래의 형태 또는 전형적인 예, 기초 또는 표준
- 시제품이 나오기 전의 제품의 원형으로 개발 검증과 양산의 검증을 거쳐야 시제품이 됨
- 정보시스템의 미완성 버전 또는 중요한 기능들이 포함된 시스템 초기 모델

프로그래밍 및 프로토타입 도구

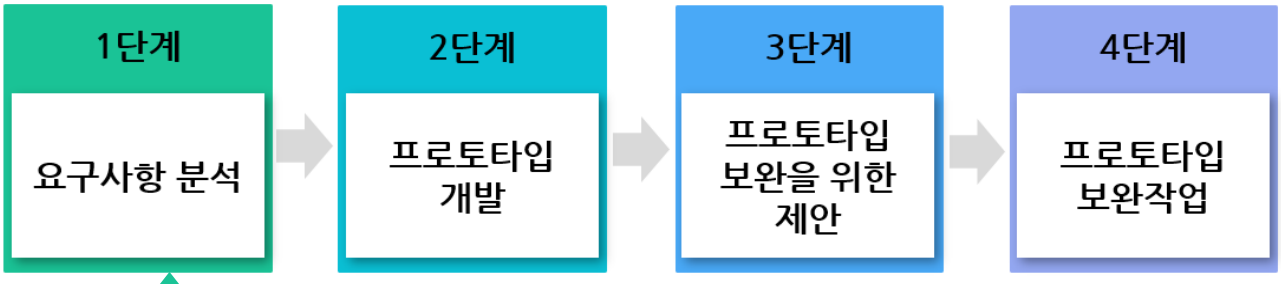
2 프로토 타입 도구

1 프로토타입(Prototype)

2 프로토 타이핑



2 프로토타이핑 단계

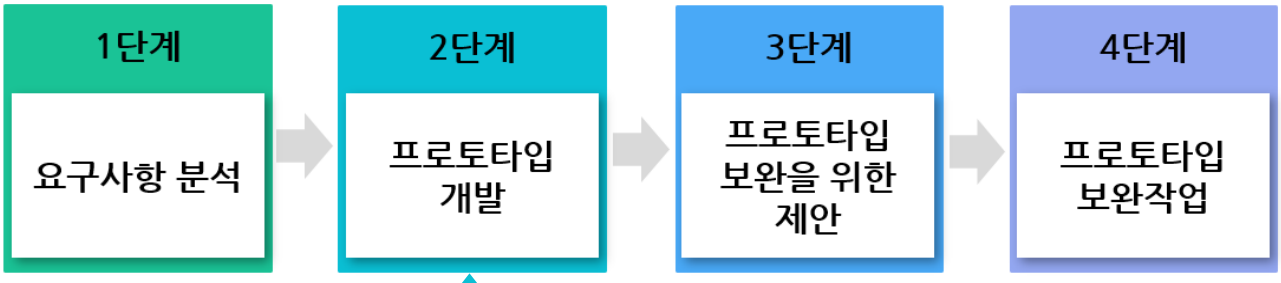


- 사용자 요구사항 분석
- 시스템 설계자는 기본적인 요구사항이 도출되기까지 사용자와 함께 작업

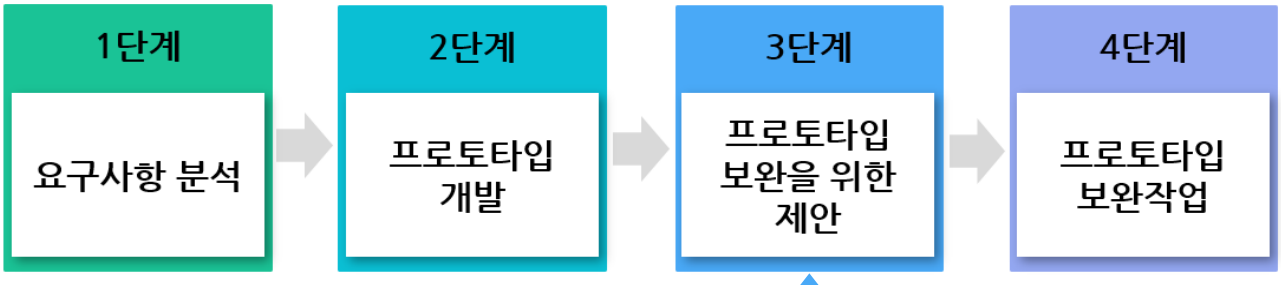
프로그래밍 및 프로토타입 도구

2 프로토 타입 도구

2 프로토타이핑 단계



- 도출된 요구사항을 만족하게 하는 프로토타입을 프로그래밍 언어 또는 CASE 도구를 이용하여 개발
- 프로토타입은 앞으로 개발될 시스템의 가장 핵심적인 기능 위주로 개발



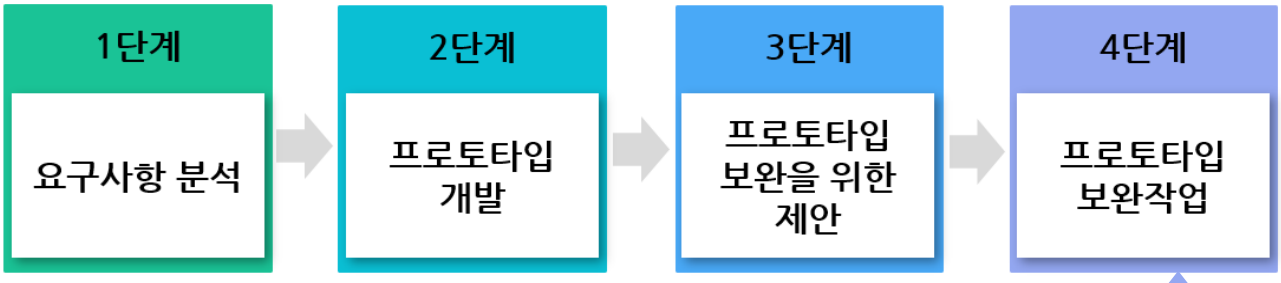
- 개발된 프로토타입을 실제 사용하여 요구사항이 반영되었는지 확인
- 새로운 요구사항에 대한 보완작업 제안

프로그래밍 및 프로토타입 도구

2 프로토 타입 도구

2 프로토타이핑 단계

2 프로토 타이핑



- 프로토타입의 수정과 보완작업 수행
- 사용자가 만족할 때까지 3, 4단계를 반복 수행

3 장점과 단점

1 장점

1. 사용자 중심 개발 방법으로 최종 사용자의 요구를 극대화함
2. 개발 시간을 줄일 수 있음
3. 오류를 초기에 발견 가능
4. 변경이 쉬움

프로그래밍 및 프로토타입 도구

2 프로토 타입 도구

3 장점과 단점

2 단점

1. 시스템 유지보수에 필수적인 시스템 문서화 과정이 지나치게 축소되거나 생략됨
2. 유지보수 시 관련 문서가 없기 때문에 불필요한 시간이 낭비됨
3. 최종적으로 시간과 비용이 많이 들 수 있음

4 프로토 타입 도구 선택 방법

01 도구를 사용하고자 하는 **목적**

02 도구가 가진 **기능**

03 도구를 배우기 위한 **난이도**

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

1 소프트웨어 품질

1 개요

소프트웨어 품질

- 소프트웨어의 바람직한 속성의 정도 및 사용자의 요구사항 기능 확보 정도
- 개발 소프트웨어의 안전성 및 기능 만족성을 평가하는 척도

2 소프트웨어 품질 3대 요소

- 투명성
- 원칙
- 국제 표준

3 소프트웨어 품질의 기대 효과

TCO : Total Cost of Ownership(총 소유비용)

1. 내부 조직의 효율화
2. 관리 TCO 절감
3. 대외 이미지
4. 신뢰도
5. 고객 만족
6. 마케팅 요소 효과

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

2 소프트웨어 품질 평가 관점

프로세스 중심평가 | 제품 중심 평가 관점 | 사용자 평가 관점 | 개발자 평가 관점 | 관리자 평가 관점

- 정립된 기준은 여러 프로세스에 적용할 수 있고 단시간 내에 평가 완료 가능
- 소규모 회사에 적용, 제품 자체의 품질 평가가 어려움

프로세스 중심평가 | 제품 중심 평가 관점 | 사용자 평가 관점 | 개발자 평가 관점 | 관리자 평가 관점

- 품질의 전문성, 객관성이 가능하고 제품 자체의 품질에 직접적인 영향을 줌
- 평가 비용, 시간, 노력이 필요

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

2 소프트웨어 품질 평가 관점

프로세스 중심평가 | 제품 중심 평가 관점 | **사용자 평가 관점** | 개발자 평가 관점 | 관리자 평가 관점

- 소프트웨어 내부보다는 사용성, 성능, 사용 효과 등의 평가

프로세스 중심평가 | 제품 중심 평가 관점 | 사용자 평가 관점 | **개발자 평가 관점** | 관리자 평가 관점

- 개발 전 과정의 단계별 다른 각도의 평가 척도가 필요

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

2 소프트웨어 품질 평가 관점

프로세스 중심평가 | 제품 중심 평가 관점 | 사용자 평가 관점 | 개발자 평가 관점 | 관리자 평가 관점

- 전반적인 품질과 일정, 비용 등의 관리 기준의 품질 평가

3 소프트웨어 품질 특성

기능성(Functionality)

- 규정된 기능의 특성, 속성
- 정확성, 적합성, 상호호환성, 유연성, 보안성

신뢰성(Reliability)

- 명시된 조건에서 주어진 성능을 유지하는 소프트웨어 능력과 그 속성
- 성숙성, 오류 허용성, 회복성

사용성(Usability)

- 사용에 대한 노력과 사용자의 심사 및 평가와 관련된 속성
- 이해성, 운용성, 습득성

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

3 소프트웨어 품질 특성

효율성(Efficiency)	유지보수성 (Maintainability)	이식성(Portability)
• 규정된 조건에서 소프트웨어 성능 수준과 자료의 사용성에 대한 속성 • 실행 효율성, 자원 효율성	• 규정된 수정사항의 수행 노력과 결과의 척도 속성 • 해석성, 안전성, 변경 용이성, 시험성	• 다른 환경으로의 이전에 관련된 소프트웨어 능력과 관련된 속성 • 적응성, 일치성, 이식 작업성, 치환성

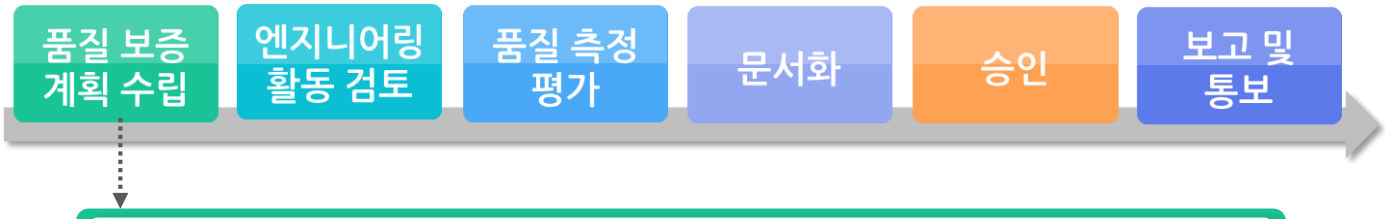
4 소프트웨어 품질 보증 기법

워크스루 (Walkthrough)	• 비공식적 검토과정, 개발참여자 위주
검토 (Review)	• 부적절한 정보, 누락 정보 발견 • 개발자, 관리자, 사용자, 외부전문가 참여
검사 (Inspection)	• 구성요소의 공식적인 정확성 평가 • 체크리스트를 이용한 전문가들에 의함

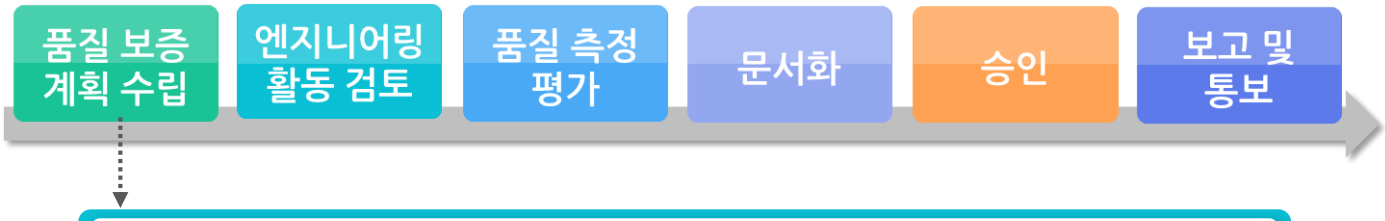
품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

5 품질 보증 절차



- 품질 보증 활동 계획 수립 및 평가 대상 산출물 설정
- 품질 보증 프로세스와 기준선 설정

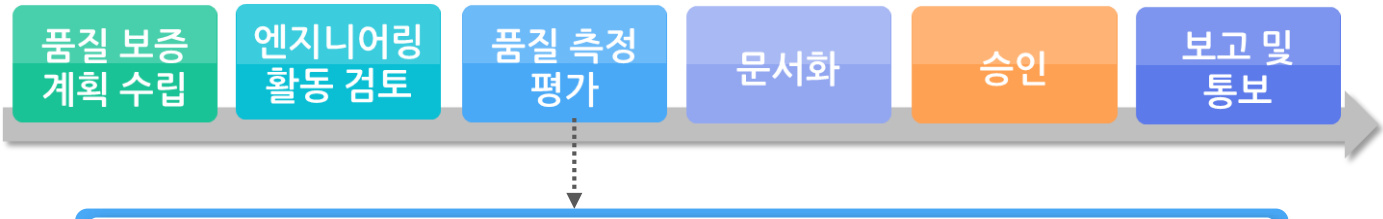


- 개발 활동에 대한 검토
- 산출물을 생산하기 위한 프로세스들의 운용 검토

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

5 품질 보증 절차



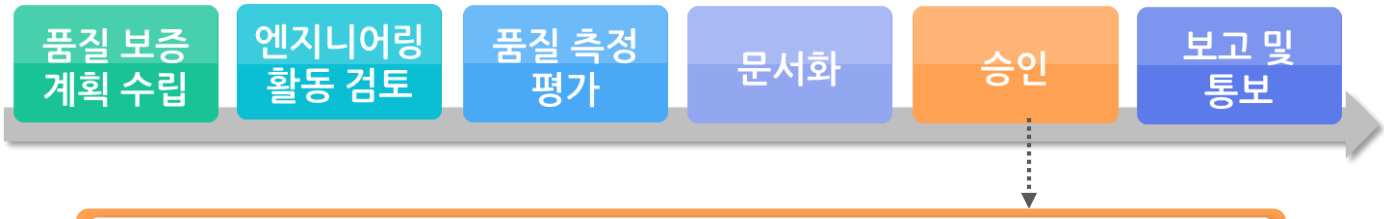
- 품질 목표에 따라 실제 품질 평가 및 측정
- 소프트웨어 감리 및 감사와 연관

- 품질 평가에 대한 문서 기록

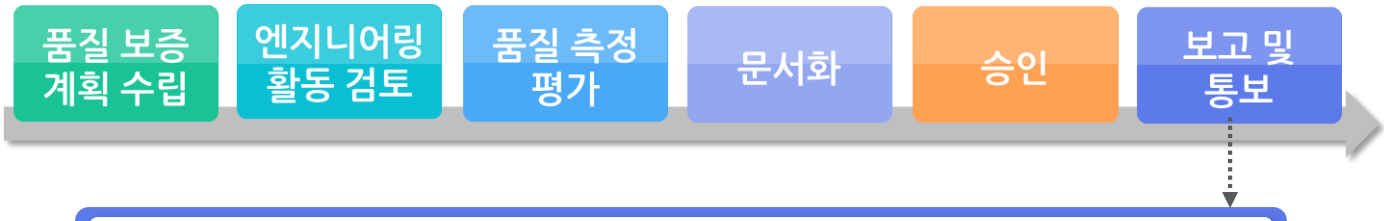
품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

5 품질 보증 절차



- 문서화된 평가 결과 승인
- 품질 보증 활동에 대한 최고 결정권자의 승인



- 승인된 품질 평가 결과를 개발 활동에 반영
- 관련 조직 및 관련 인원에 통보

품질 보증 및 유지보수 도구

1 소프트웨어 품질 보증

6 소프트웨어 품질 보증 활동

형상 관리	형상 항목 식별, 변경 사항 관리
문서 관리	문서 관리 절차 수립, 문서 작성 / 보관 / 폐기
품질 기록	품질 보증 계획 / 수행 / 결과 기록
합동 검토	마일스톤에 따라 프로젝트의 진행 상황을 공동 검토
검증 및 확인	단계별 검증 및 테스트
시정 조치	해결 방안 수립 및 조치
위험 관리	예상 위험 발견 / 평가 / 통제
쟁점 관리	고객 요구사항 변경 등의 쟁점 분석 대안 설정 및 실행

품질 보증 및 유지보수 도구

2 소프트웨어 유지보수

1 소프트웨어 유지보수

1 개요

소프트웨어 유지보수란?

- SDLC의 마지막 단계로 소프트웨어의 생명을 연장하는 운영 중심의 작업 단계
- 오류의 수정, 요구사항 정정, 기능과 수행력을 증진시키는 일련의 작업
- 소프트웨어가 납품된 후 결함의 제거, 성능 향상, 변환된 환경에 적응토록 처리

2 목적

01

소프트웨어의
성능 개선, 하자 보수

02

새로운 환경에서
동작할 수 있도록
이식 및 수정과 일련의 예방적 조치

품질 보증 및 유지보수 도구

2 소프트웨어 유지보수

1 소프트웨어 유지보수

3 유지보수의 중요성

- 소프트웨어 예산에서 기존 개발비용보다 많은 유지보수 비용의 급증
- 소프트웨어 복잡화에 따른 난해함으로 문서화 등의 관리업무 증가
- 개발 기간보다 사용 기간이 길기 때문에 더욱 관리의 중요성이 강조
- 용역 개발보다 패키지의 선택이 확산됨에 따라 유지보수 부문이 증가함

2 소프트웨어 유지보수 순서

품질 보증 및 유지보수 도구

2 소프트웨어 유지보수

3 유지보수 발생 요인

소프트웨어의 유기적인 측면

- 사용자 참여 및 만족도 미흡, 시스템의 문서화, 유연성, 신뢰성 저하

개발 환경적 측면

- 유지보수보다는 신규 프로젝트에 치중
- 표준화된 방법론 및 개발 도구 적용의 부재
- 분석 설계 시 유지보수성 및 재사용성 요인 반영의 미흡

4 유지보수의 문제점

유지보수에 따른 부작용 발생

➡ 코딩, 자료, 문서화 부작용

시스템의 신뢰성 저하 가능성 발생,
유지보수 비용 및 인력의 증가

유지보수 요원의 기술 부족, 비체계적인 유지보수

➡ 절차, 조직, 인력 운영 방법 등이 비체계화 시 발생

품질 보증 및 유지보수 도구

2 소프트웨어 유지보수

3 유지보수 발생 요인

1. 표준화된 개발방법론 및 개발 도구(CASE 등) 적용
2. 소프트웨어 재공학 도구 활용
   - 분석, 재구조화, 역공학 등의 활용
3. SDLC 단계 중 각 단계의 TEST 및 품질 관리 강화
4. 유지보수 발생 요인에 대한 예방 활동 실시
   - 요구사항 통제, 품질 관리, 감리 등
5. 적절한 프로젝트 관리 기법의 도입
   - 형상 관리 등
6. 4세대 언어의 활용
7. Component 개념의 활용으로 재구조화, 재사용화에 대한 도입

학습정리

1. 프로그래밍 및 프로토타입 도구

- 프로그래밍 도구는 소프트웨어 개발자가 다른 프로그램과 응용 프로그램을 만들고 오류를 고치고 유지보수하는 데에 사용하는 프로그램이나 응용 프로그램
- Unix 시스템의 기본 개발 도구로 시스템에 패키지로 추가된 vi, vim이 있음
- C 언어 개발을 위한 가장 대중적인 프로그램으로는 Microsoft Visual Studio를 많이 사용하며, 최근 반응형 웹페이지 개발에 도움이 되는 기능도 제공하고 있음(Express 버전으로 제한된 기능 제공)
- Java 언어 개발을 위한 가장 대중적인 프로그램으로 eclipse를 사용하며, 서버 쪽 개발, 형상 관리, 데이터베이스 연동이 쉬우며 plugin을 통해 방대한 기능을 제공함
- Forntend 개발용 WebStrom은 Git을 이용한 형상 관리 및 CSS, HTML 자동 완성기능을 제공하며 서버 쪽 개발에도 여러 가지 편리한 기능을 제공함
- 프로토타입 도구의 선택할 때 사용하고자 하는 목적, 도구가 가진 기능, 도구를 배우기 위한 난이도를 고려하여 도구를 선택함

학습정리

2. 품질 보증 및 유지보수 도구

- 소프트웨어 품질은 개발 소프트웨어의 안전성 및 기능 만족성을 평가하는 척도임
- 소프트웨어의 품질의 3대 요소로는 투명성, 원칙, 국제 표준이 있음
- 소프트웨어 품질의 평가 관점은 프로세스 중심 평가, 제품 중심 평가 관점, 사용자 평가 관점, 사용자 평가 관점, 개발자 평가 관점, 관리자 평가 관점으로 나눌 수 있음
- 소프트웨어 품질 보증 기법으로 비공식적 검토과정으로 워크스루 기법이 있으며, 부적절한 정보 및 누락 정보를 발견하는 검토 기법, 구성요소의 공식적인 정확성을 평가하는 검사 기법이 있음
- 소프트웨어의 유지보수 목적은 성능을 개선하고 하자를 보수하며 새로운 환경에서 동작할 수 있도록 이식 및 수정과 일련의 예방적 조치를 목적으로 함