1강. 소프트웨어 공학 개요

컴퓨터과학과 김희천 교수

목차

Chapter. 1

소프트웨어의 정의와 성질

1. 소프트웨어

소프트웨어 정의

- 좁은 의미의 소프트웨어는 프로그램과 관련 데이터의 묶음
- 포괄적 의미의 소프트웨어는 관련 문서들을 포함한 개념

소프트웨어의 중요성과 역할

- 사업체의 의사결정과 과학적/공학적 문제 해결의 지원 도구
- 모든 종류의 컴퓨터 시스템에 내장되어 사용됨
  - 모든 산업의 기반 기술로서 중요성이 증가
  - 세상을 바꾸는 원동력

2. 소프트웨어의 분류 (1/2)

기능에 따른 분류

시스템 소프트웨어

- 컴퓨터를 동작 시키기 위한 목적의 소프트웨어
  - 운영체제, 장치 드라이버
  - 컴파일러, DBMS
  - 유틸리티 프로그램 등

응용 소프트웨어

- 사용자의 실제 업무를 수행하는 소프트웨어
  - 웹 브라우저
  - 사무용 SW, 게임 SW
  - MIS, ERP 등

2. 소프트웨어의 분류 (2/2)

사용자에 따른 분류

일반(generic) 소프트웨어

- 불특정 다수를 대상으로 설계되는 패키지 소프트웨어
- 요구사항이 일반적이고 안정적
- 상용 제품으로 판매될 수 있음
- 데이터베이스 관련 제품, 사무용 패키지, 운영체제 등

맞춤형(custom) 소프트웨어

- 특정 고객을 위해 주문 제작됨
- 응용 도메인, 사용 환경, 요구사항이 특별함
- 프로세스 제어, 교통 관제 시스템, 병원 관리 시스템 등

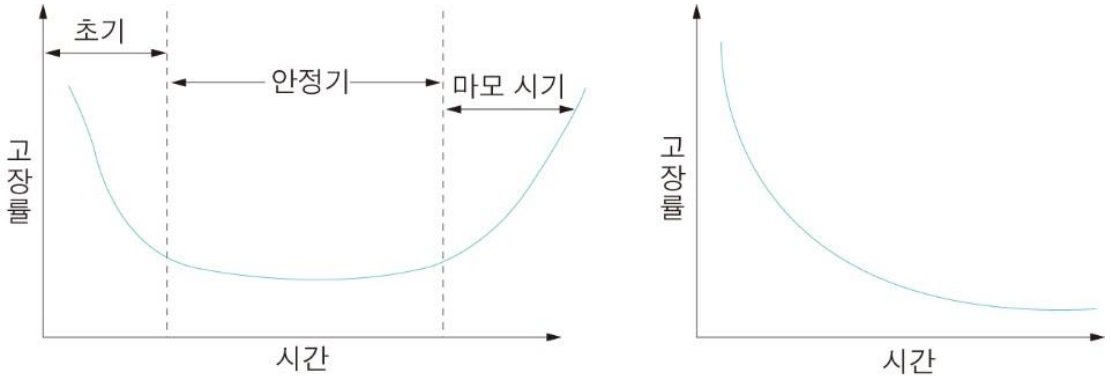
3. 소프트웨어의 성질 (1/2)

- 무형의 인공물로 물질적인 성질이 없음
- 컴포넌트들의 조립을 통해 만들기 어려움
- 상대적으로 설계 과정의 품질 보증 활동이 중요함
- 개발 비용의 대부분은 노동력에 투입됨
- 상대적으로 변경이 용이함
  - 소프트웨어의 유연성 또는 순응성이라고 함

3. 소프트웨어의 성질 (1/2)

✚ 소프트웨어는 마모되지 않음

- ✖환경이 변화하여 쓸모가 없어지거나 품질이 저하될 수 있을 뿐
- ✖하드웨어 신뢰성을 보여주는 욕조 곡선(bathtub curve)과 비교됨



✚ 소프트웨어 유지보수 작업은 설계의 변경을 요구함

4. 소프트웨어 응용 분야 (1/2)

시스템 소프트웨어

실시간 소프트웨어

- 이벤트 발생과 처리가 실시간으로 이루어짐
- 자동 제어 시스템, 은행 시스템, 좌석 예약 시스템

내장형 소프트웨어

- 시스템의 일부로 내장된 소프트웨어
- 자동차, 전자 제품 등에 내장된 제어 소프트웨어

비즈니스 소프트웨어

- 사업 목적의 업무를 처리
- 회계 업무 패키지, 급여 관리나 재고 관리 패키지, 경영 정보 시스템 등

4. 소프트웨어 응용 분야 (2/2)

개인용 소프트웨어

- 워드, 멀티미디어나 게임 소프트웨어

인공지능 소프트웨어

- 복잡한 문제 해결을 위한 것
- 로보틱스, 전문가 시스템, 화상/음성 인식 소프트웨어

웹 기반 소프트웨어

- 웹 브라우저를 통해 요청되고 수행되는 소프트웨어

공학용/과학용 소프트웨어

- CAD, 기상 분석이나 유전자 분석용 소프트웨어

Chapter. 2

소프트웨어 공학의 정의

1. 소프트웨어 위기

- 1968년 NATO 소프트웨어 공학 컨퍼런스
- 하드웨어 기술의 급격한 발전과 요구의 다양화와 복잡화
  - 규모가 크고 복잡한 소프트웨어를 빠르게 개발해야 할 필요성
- 그러나 소프트웨어의 기술 발전이 더딤을 일컫는 용어
- 소프트웨어 공학이라는 학문 분야가 등장하게 된 배경

2. 소프트웨어 위기 현상의 사례

- 개발 일정의 지연
- 초과 비용의 발생
- 제품 신뢰도의 결여
- 명세를 충족하지 못하는 제품
- 유지보수의 어려움

3. 소프트웨어 위기 현상의 원인

- 소프트웨어 엔지니어의 부족
- 경영층의 인식 부족
- 방법론과 도구의 부재, 개발 생산성의 저하
- 소프트웨어 자체의 복잡성 증가

4. 소프트웨어 공학의 유래

- 공학이란 문제 해결을 위한 재료/구조물/기계 등의 결과물을 생산하기 위해 지식이나 경험의 적용을 연구하는 분야
- 소프트웨어 공학은 1968년 NATO 소프트웨어 공학 컨퍼런스 이후로 고품질 소프트웨어의 경제적이고 빠른 생산과 유지보수를 위한 연구 분야가 됨
- 소프트웨어 위기 현상을 부각하고 해결책으로 표현하기 위해 만든 용어

5. 소프트웨어의 공학의 정의

NATO 컨퍼런스의 바우어 교수

"신뢰성 있고 요구기능을 효율적으로 수행하는
소프트웨어를 경제적으로 생산하기 위해 건전한 공학적
원리와 방법을 만들고 사용하는 것"

IEEE 소프트웨어 공학 표준 용어집

"소프트웨어의 개발, 운영, 유지보수에 체계적이고
제어가능하며 정량화된 접근 방법을 적용하는 것,
즉 소프트웨어 개발에 공학 기술을 적용하는 것"

기타

"인간에게 유용한 소프트웨어 제품을 만드는 과정에
과학적 지식을 적용함으로써 실제적
문제의 비용 효율적 해결책을 다루는 일"

Chapter. 3

소프트웨어 공학 환경

1. 소프트웨어 개발 시 고려 사항

소프트웨어 개발이 단순한 코드의 생성이 아님

- 요구사항 명세, 설계, 프로젝트 관리 등 통합적 문제 인식이 필요

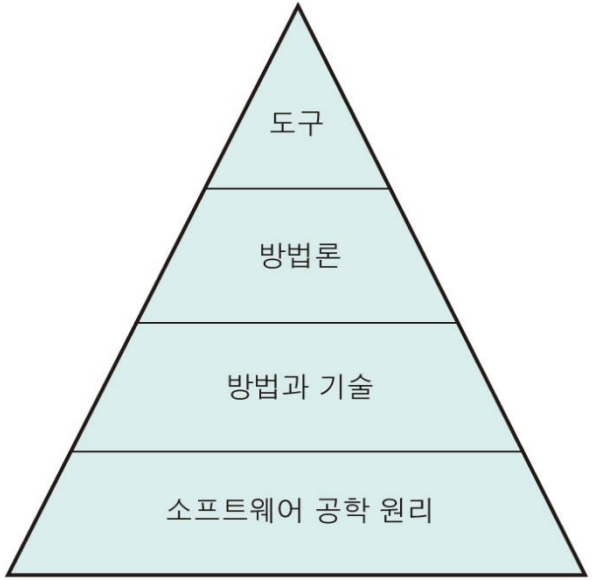
개발 프로세스와 프로젝트 관리가 중요함

비용을 줄이고 제품의 품질을 높이기 위한 방법은?

- 문서화와 프로젝트 관리가 중요함
- 초기 요구사항 명세화 작업에 노력을 기울여야 함
- 변경이나 재사용을 염두에 둔 작업이 필요함

2. 소프트웨어 공학 환경

- 소프트웨어 공학의 대상은 중규모 이상의 복잡하고 중요한 소프트웨어
- 다양한 해결 방법들을 통합적으로 다루어야 함
- 소프트웨어 공학 환경의 구성
  - 계층적 표현
  - 소프트웨어 공학 원리에 기초하여 방법과 기술이, 방법과 기술을 가지고 방법론이, 방법론에 기초하여 도구가 만들어짐



3. 소프트웨어 공학 환경의 구성

소프트웨어 공학 원리

- 소프트웨어 프로세스와 제품의 바람직한 측면들을 기술하는 일반적이고 추상적인 설명
- 예) 추상화, 분할정복, 계층적 조직의 원리

방법과 기술

- 행위를 통제하는 체계적이고 일반적인 가이드라인
- 바람직한 속성을 프로세스나 제품에 포함시키기 위해 필요

방법론

- 방법과 기술들의 조합으로 문제 해결을 위해 정해진 프로세스 안에서 조직화 한 것
- 프로세스(what)와 방법(how)을 함께 기술한 것

4. 소프트웨어 프로세스 모델

- 소프트웨어의 생산과 진화 과정을 추상화하여 요약 표현한 것
  - 생명주기 모델
- 시스템 개념화, 요구사항 정의, 설계, 그리고 구현에 이르는 전이 과정을 표현
  - 필요한 활동들, 결과물, 사람의 역할 등을 포함
- 좋은 프로세스 모델은
  - 전이 과정에서 생기는 문제를 최소화해야 함
  - 공통 개발 프레임워크를 제공하여 생산성을 향상시킴
  - 개발자 간 공통의 문화와 공통의 기술을 제공함

Chapter. 4

좋은 소프트웨어의 기준

1. 외부 품질과 내부 품질

외부 품질

- 사용자가 인지할 수 있는 품질 요소
- 사용성, 신뢰도, 속도

내부 품질

- 외부 품질의 향상에 도움을 주는 것으로 개발자에게 중요함
- 잘 작성된 요구사항이나 설계 문서

2. 소프트웨어 신뢰도 (1/2)

- 사용자가 소프트웨어를 신뢰하는 정도
- 정확한 결과를 적시에 제공
- 오랜 시간 작동되며 필요할 때 사용할 수 있음
- 오류 발생이 적고, 오류가 발생 후에 무난히 복구되며, 오류 결과가 치명적이지 않으며 강건함

2. 소프트웨어 신뢰도 (2/2)

- 시스템의 종류나 특성에 따라 측정 방법이 다름
  - 주어진 시간 동안 정확히 작동될 확률
  - 고장의 빈도, 평균 수명 등
- 시간이 지남에 따라 안정화되나, 소프트웨어 고장 함수가 욕조 곡선과 유사한 형태를 가지는 경우도 있음
  - 버그 수정이나 기능의 개선이나 추가로 인해 새로운 오류가 유입되는 경우
  - 하드웨어나 운영체제를 변경하는 경우
  - 적정 용량이나 성능을 초과하여 사용하는 경우

3. 정확성, 성능, 사용성

소프트웨어의 정확성

- 명세서와 일치하게 작동하는 능력, 사용자의 요구를 만족시키는 능력
- 작은 결함이 있어도 정확한 것은 아니나 신뢰도에 문제가 되지 않을 수 있음

소프트웨어의 성능

- 지정된 시간 안에 컴퓨터 시스템이 처리할 수 있는 작업량
- 주어진 조건 하에서 주어진 기능을 빠르게 제공하는 능력

소프트웨어의 사용성

- 본래의 목적으로 효율성 있게 사용할 수 있는가

4. 상호운영성, 유지보수성

소프트웨어의 상호운영성

- 다른 시스템과 공존하며 협력할 수 있는 능력

소프트웨어의 유지보수성

- 소프트웨어의 변경이 용이한 정도
- 요지보수 요인
  - 새로운 기능의 추가나 기존 기능의 개선
  - 환경의 변화에 따른 수정, 존재하는 오류의 수정

5. 이식성, 검사성, 추적성

소프트웨어의 이식성

- 다른 환경에서 동작할 수 있는 능력
- 이식 요인
  - 하드웨어, 운영체제, 상호작용하는 시스템의 변경 등

소프트웨어의 검사성

- 좋은 소프트웨어의 속성을 갖추었는지 검사할 수 있는가

소프트웨어의 추적성

- 이해관계자(stakeholder), 요구사항, 설계문서,
  소스코드 간의 관계를 추적할 수 있는가

다음강의

2강. 소프트웨어 프로세스