1주차
2차시

02

소프트웨어공학

소프트웨어 위기와 공학의 개념

1 소프트웨어 위기

2 소프트웨어 공학 개요

01
소프트웨어 위기

1. S/W의 위기(crisis)

- S/W 생산성을 높일 수 있는 기술과 전문 인력이 절대적으로 필요한 반면 공급이 수요를 충족하지 못해서 S/W에 대한 사용자들의 요구사항을 처리할 수 없는 문제가 발생한 것

2000년 이전	하드웨어가 급속하게 발전하고, 컴퓨터가 대중화 되었다. SW 개발 및 생산 활동은 매우 저조했다
2000년 이후	SW 전문가들이 늘어나 생산성이 높아졌다. 사용자의 요구사항 및 품질에 대한 기대감이 계속 증가하여 SW 위기 발생 요인은 계속되고 있다.

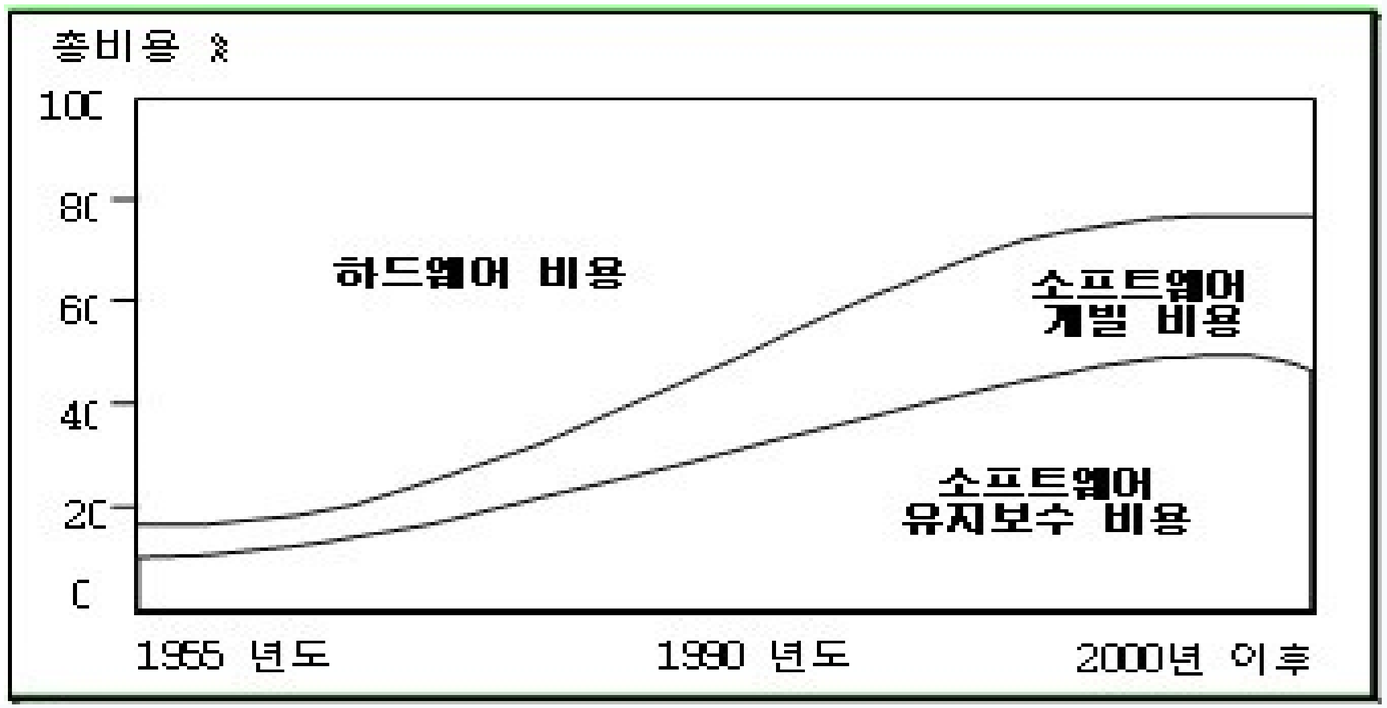
SW 위기 발생 요인	SW 위기 결과
① S/W 규모의 증대와 복잡도에 따른 비용 증가 ② 프로젝트 관리기술의 부재 ③ S/W 개발기술에 대한 교육/훈련 부족 ④ S/W 품질의 미흡 ⑤ S/W 생산성 저조	① 유지 보수가 어렵다. ② 성능 및 신뢰성이 부족. ③ 개발 기간 지연 및 개발 비용의 증가로 연계 ④ 개발 전문가 부족과 이로 인한 인건비 상승

01
소프트웨어 위기

1. S/W의 위기(crisis)

01
소프트웨어 위기

2. S/W의 비용 구조



01
소프트웨어 위기

3. 소프트웨어 복잡성

- 소프트웨어 복잡성의 증가
  - 사용자들이 복잡하고 고급스러운 기능 요구
    - ✓ 다양한 글씨체, 철자 검색, 문법 검색 등
  - 이질적인 전산 환경
  - 분산 처리의 보편화

01
소프트웨어 위기

4. 복잡한 소프트웨어 개발

- 단순 확대 적용 문제가 아님
  - 많은 인력 및 충분한 시간 투입

- 비유
  - 시냇물 위에 통나무 다리를 놓는 일 → 한강 남북을 연결하는 다리를 놓는 일

- 문제의 원인
  - 소프트웨어의 복잡성

01
소프트웨어 위기

5. 소프트웨어 비용

- 개발 비용 및 운용 비용
  - 소프트웨어 운용 비용
  - 소프트웨어 운용에 필요한 일체의 경비
  - 유지보수와 기능 보완 등의 비용

01
소프트웨어 위기

6. S/W의 위기(crisis) 해결 방안

- 단일 방법론으로 해결되지 않아 다양한 방법론을 혼합하여 해결

- SW 개발 측면
  - 표준화된 방법론 적용, 객체지향 프로그래밍, 구조적 프로그래밍, SW 컴포넌트화, SW 프로토타이핑, 반복형 개발, 버그/ 이슈 관리시스템, 버전 관리 시스템, 품질보증 기법 적용, 통합개발 환경, 애자일 개발 프로세스 등을 프로젝트 성격에 맞게 적용

- SW 유지보수 측면
  - SW 재사용, 객체지향기술, 캡슐화, CBD(Component Based Development) 응용, 디자인 패턴(Model + View + Controller),가비지 컬렉션, 멀티쓰레드 프로그래밍 등을 활용

02
소프트웨어 공학 개요

1. SW 공학이란?

- 인간에게 필요한 프로그램과 프로그램의 개발, 운용, 유지보수에 필요한 관련 정보 일체를 의미하고, 여러 가지 방법론과 도구, 관리 기법을 통하여 S/W 품질과 생산성 향상을 목적

- 공학적 원리에 의하여 SW를 개발하는 학문으로 SW 제품의 생산, 유지보수 관련 기술, 경영에 관한 학문

- 과학적 지식을 컴퓨터 프로그램 설계와 제작에 실제 응용하는 것이며, 이를 개발 운영하고 유지 보수하는데 필요한 문서화 작성 과정을 포함

02
소프트웨어 공학 개요

1. SW 공학이란?

- **소프트웨어 공학의 다양한 정의**
  - IEEE[전기전자공학자협회: Institute of Electrical and Electronics Engineers]
    - ✓ 소프트웨어의 개발과 운용 및 유지보수에 대한 체계적(systematic)이며 훈련된 (disciplined) 계량적(quantifiable) 접근 방식의 적용
    - ✓ 소프트웨어에 대한 공학적 접근 적용
  - Fritz Bauer
    - ✓ 컴퓨터 H/W에서 신뢰성 있게 운영되는 S/W를 경제성 있게 개발하기 위해 공학적 원리를 응용하고 확립 시킨 이론

02
소프트웨어 공학 개요

1. SW 공학이란?

- **소프트웨어 공학의 다양한 정의**
  - Berry Boehm
    - ✓ 컴퓨터 프로그램의 설계, 개발, 운영, 유지보수와 관련하여 문서를 작성하는데 필요한 과학적 지식의 실용화
  - Richard R. Fairley
    - ✓ 전산학, 경제학, 경영과학 및 의사소통기술과 문제해결을 위한 공학적인 접근방안을 토대로 소프트웨어 개발에 임하는 신기술 체계

02
소프트웨어 공학 개요

2. SW 공학의 영역

분 야	의 미	사 례	주택 건축 비유
방법 (method)	S/W 제작에 사용하는 기법이나 절차	- 구조적 분석 - 객체지향 분석 - 설계 방법	목조주택, 벽돌, 조립식, 철근콘크리트 등
도구 (tool)	자동화된 시스템	- 설계 도구 - 프로그래밍 도구 - 테스트 도구	콘크리트 드릴, 자동 톱 철근 절단기 등
프로세스 (process)	도구와 기법을 사용하여 작업하는 순서	- unified process - eXtreme programming	건축 시공 순서
패러다임 (paradigm)	접근 방법, 스타일	- 구조적 방법론 - 객체지향 방법론	한옥식, 유럽식, 중국식 퓨전 등

02
소프트웨어 공학 개요

3. SW 공학의 목표 (QCD의 만족, Quality, Cost, Delivery)

- 오류가 없는 고품질(Quality) 소프트웨어를 계획된 개발기간을 넘기지 않고, 추가적인 예산의 요청 없이 사용자가 원하는 기능을 유지보수가 용이하도록 복원력이 좋은 소프트웨어를 개발하는 것.

목 표	필요 기법
고품질(Quality) 소프트웨어의 생산	요구사항 관리, 품질관리
정해진 비용, 기간, 자원으로 소프트웨어 생산	요구사항 관리, 프로젝트 관리
사용자 만족도 증진	요구사항 관리, 품질관리
소프트웨어 생산 프로세스 수행능력 개선	요구사항 관리, 적정한 SDLC
생산성(Productivity) 향상	요구사항 관리, 부품화, 모듈화, 패턴화 기법

02
소프트웨어 공학 개요

4. 기본 원칙

- 현대적인 프로그래밍 기술을 지속적 적용
- S/W 품질이 유지되도록 지속적인 검증을 시행
- S/W 개발 관련 사항 및 결과에 대한 명확한 기록을 유지

02
소프트웨어 공학 개요

5. Good S/W 특성

- 유지보수가 용이해야 한다.
- 사용자가 원하는 대로 정확히 동작해야 한다.
- 신뢰성이 높아야 하며, 효율적이어야 한다.
- 잠재적인 에러(error)가 가능한 한 적어야 한다.
- 사용하기 쉬워야 한다.

02
소프트웨어 공학 개요

5. Good S/W 특성

- 문서화가 잘 되어 있어야 한다.
- 연관된 소프트웨어 개발 시 재사용이 가능해야 한다.
- 여러 환경에서 동작될 수 있도록 이식성이 좋아야 한다.
- 적절한 사용자 인터페이스를 제공해야 한다.
- 하드웨어 자원을 효율적으로 이용할 수 있어야 한다.