
**FP 명세서를 MAN-MONTH 기준으로 작성해
오시오**

WBS를 검토합시다

Scrum의 이번 Sprint는
어떤 에픽의 스토리를 충족하기
위함인가요?

지난번 Sprint의 Post Mortem
결과가 어땠나요?

**Rational Rose로 시스템 컴포넌트
간의 Activity Diagram을 작성해 오세요**

이 교차로 모니터링 시스템은 싱글턴과
옵저버 패턴을 따르도록 디자인해주세요

이 함수의 Unit Test 진행하신건가요?
Stress Test는 언제하실 계획인가요?

CI/CD 체계 구축하셨습니까?

Nightly build가 깨져서
QA팀이 할 수 있는 것이 지금 없어요

우리의 MVP는 무엇인가요?

PMF가 적절한가요?

내일 오전 10시에 Team B랑
Data Contract 협의해요

RFP 공지되었어요

Main branch로 merge 하시죠

소통

잘 듣고
잘 받아 적고
잘 이행하고
사회성 좋고
그러면

끝????

- ~~그게 뭔가요?~~

- ~~모르겠는데요~~ 역량 고과 빵점

- ~~제 소관이 아닙니다~~

학생 vs. 소프트웨어
엔지니어

공지사항

- 3월 12, 13, 14일 휴강 --> 3월 23일 토요일 보강
1부: 9am – 12pm;
2부: 1pm – 4pm;

주의사항

- 이메일 형식

제목: [소공2] 본문의 간단한 요약

// 이름이나 학번을 굳이 제목에 넣을 필요 없음

본문:

안녕하세요 윤영 교수님,

3장 연습문제 1번에 대한 질문이 있습니다.

.

.

감사합니다,

홍길동 드림(올림)

교수



Young Yoon, Ph.D. (윤영 교수)

- Associate Professor, Department of Computer Engineering, Hongik University (부교수, 컴퓨터공학과, 홍익대학교)
- CTO and Founder of [Neouly Incorporated](#)
- [CV](#), [LinkedIn](#)

<https://apl.hongik.ac.kr>

Chap 01 소개

목 차

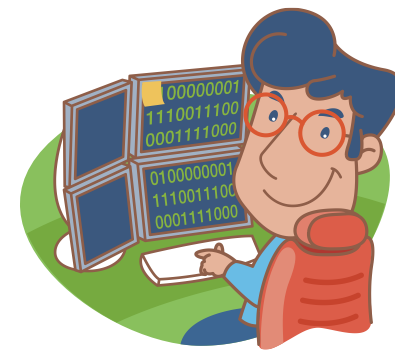
1.1 소프트웨어

1.2 소프트웨어 개발 작업

1.3 소프트웨어 공학의 접근법

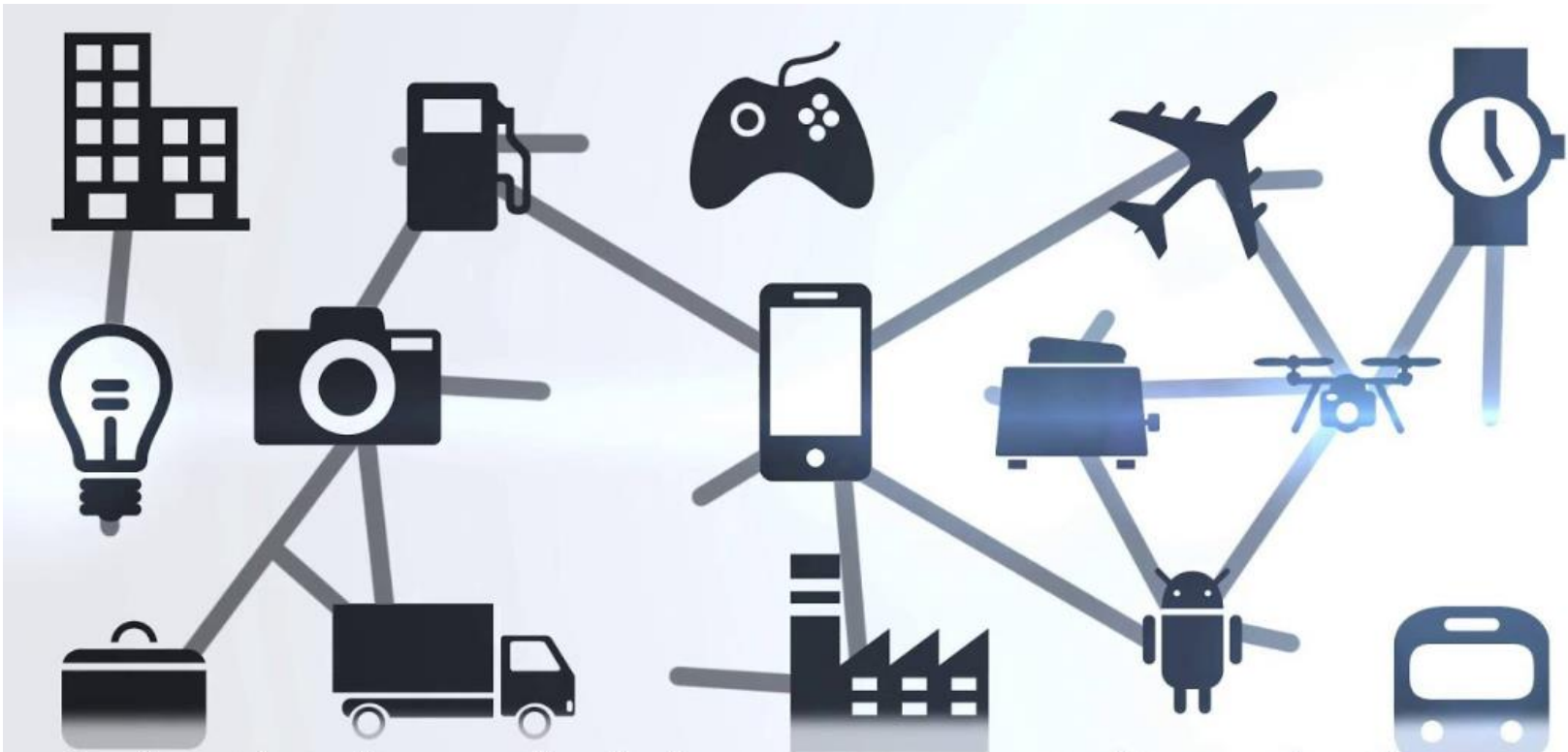
1.4 소프트웨어 공학의 주제

1.5 관련 분야



어디나 존재하는 SW




- 의존성(dependability)



1.1 소프트웨어

- 소프트웨어
 - 프로그램 + 프로그램의 개발, 운용, 보수에 필요한 정보 일체
 - 개념적이고 무형적 (생산물의 구조가 코드 안에 숨어 있음)
- 소프트웨어의 특징
 - 복잡성(complexity)
 - 순응성(conformity)
 - 변경성(changeability)
 - 비가시성(invisibility)

소프트웨어의 유형

소프트웨어 분류	특징	사용되는 카피의 수	요구되는 하드웨어 성능	개발 인력
주문형 소프트웨어 	<ul style="list-style-type: none"> 특정 고객 또는 기업의 요구를 만족시키기 위하여 제작한 소프트웨어 	적음	낮음	많음
패키지 소프트웨어 	<ul style="list-style-type: none"> 패키지화 하여 상업적으로 판매하는 소프트웨어 워드프로세서, 스프레드시트, 유통업체의 POS(Point of Sales) 시스템, 재정 분석, 주문 관리, 회계 관리 시스템 	중간	높음	중간
임베디드 소프트웨어 	<ul style="list-style-type: none"> 다른 시스템에 내장된 소프트웨어 	많음	중간	적음

소프트웨어와 시스템

- 시스템
 - 필요한 기능을 실현시키기 위하여 관련 요소를 어떤 법칙에 따라 조합한 집합체

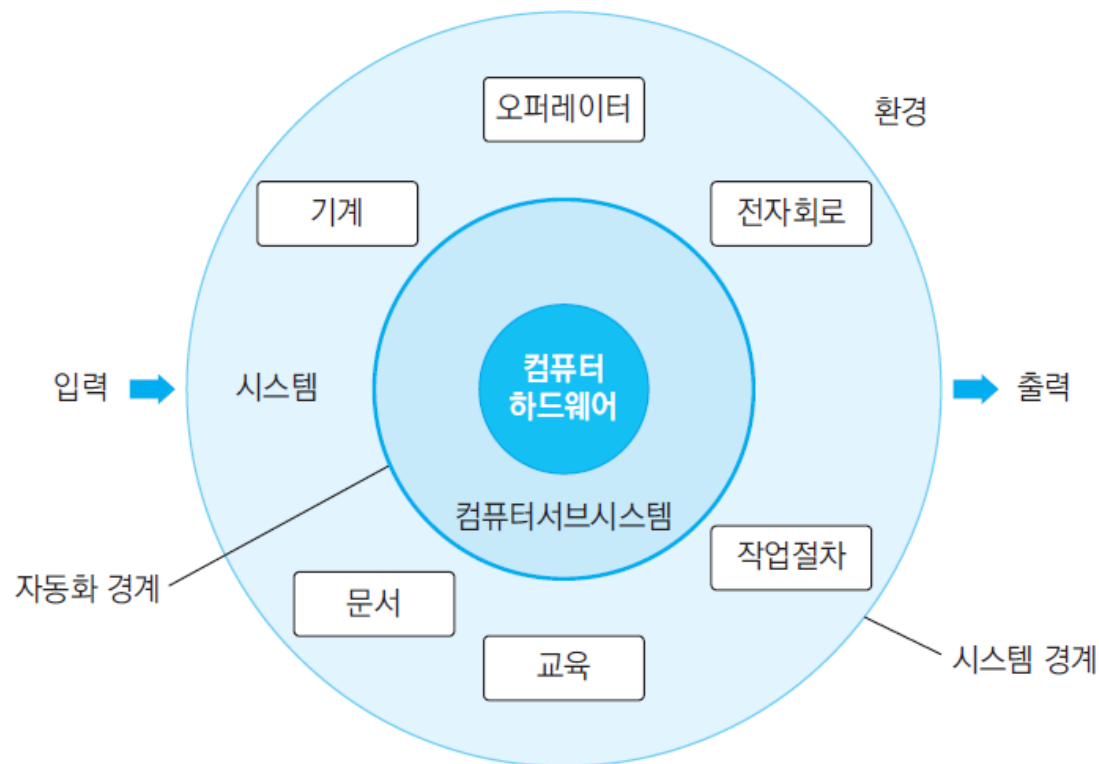


그림 1.2 시스템의 요소

1.2 소프트웨어 개발 작업

- 기본 활동

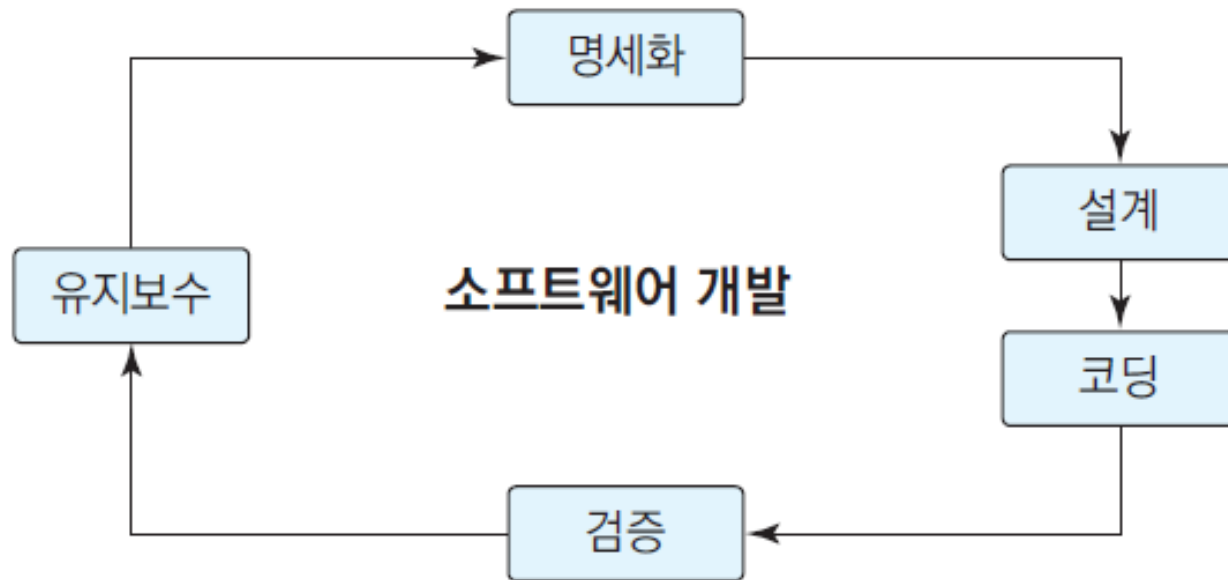


그림 1.3 소프트웨어 개발의 기본 활동

개발 작업의 특징

- 명세화의 어려움
- 재사용의 어려움
- 예측의 어려움
- 유지보수의 어려움

소프트웨어의 위기

- 소프트웨어 위기(software crisis)
 - 소프트웨어 수요가 급격히 증가하고 그 복잡성이 증가함에 따라 기존 방법이 충분하지 않아 발생한 문제

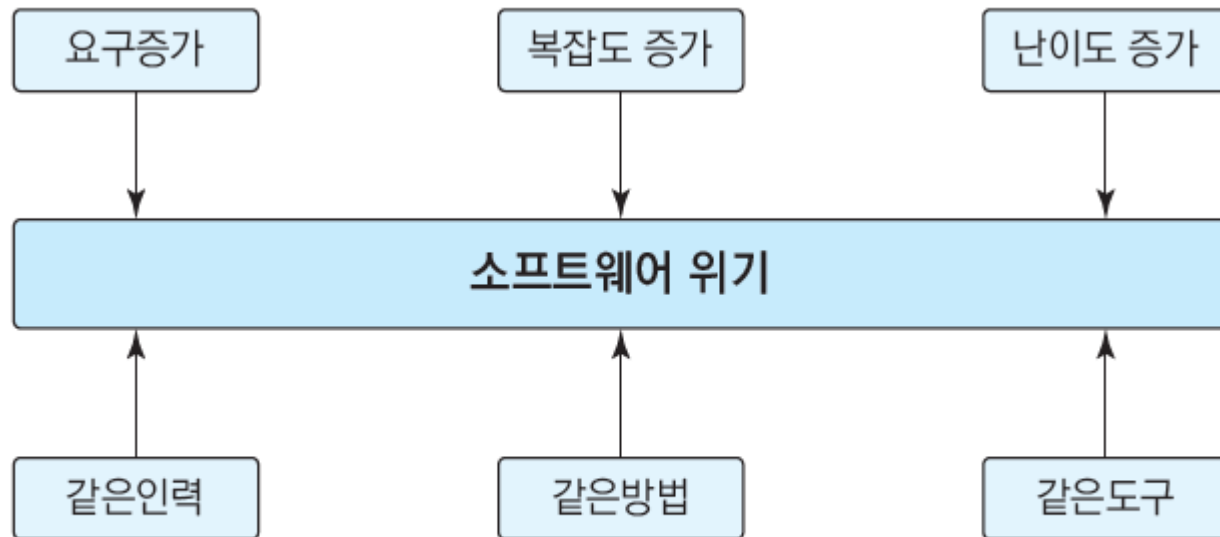


그림 1.5 소프트웨어 위기의 원인

1.3 소프트웨어 공학의 접근법

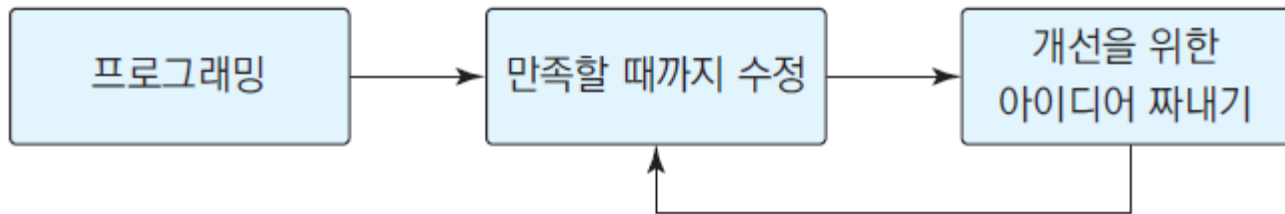


그림 1.6 즉흥적인 소프트웨어 개발

- 즉흥적인 개발의 문제점을 극복
 - 개발 지연과 예산 초과
 - 낮은 품질
 - 유지보수 곤란
 - 재작업

소프트웨어 공학의 정의

- 소프트웨어 공학
 - 소프트웨어의 개발과 운영, 유지보수, 소멸에 대한 체계적인 접근 방법
 - 소프트웨어 개발에 사용되는 방법이 일회성이 아닌 반복 사용이 가능함

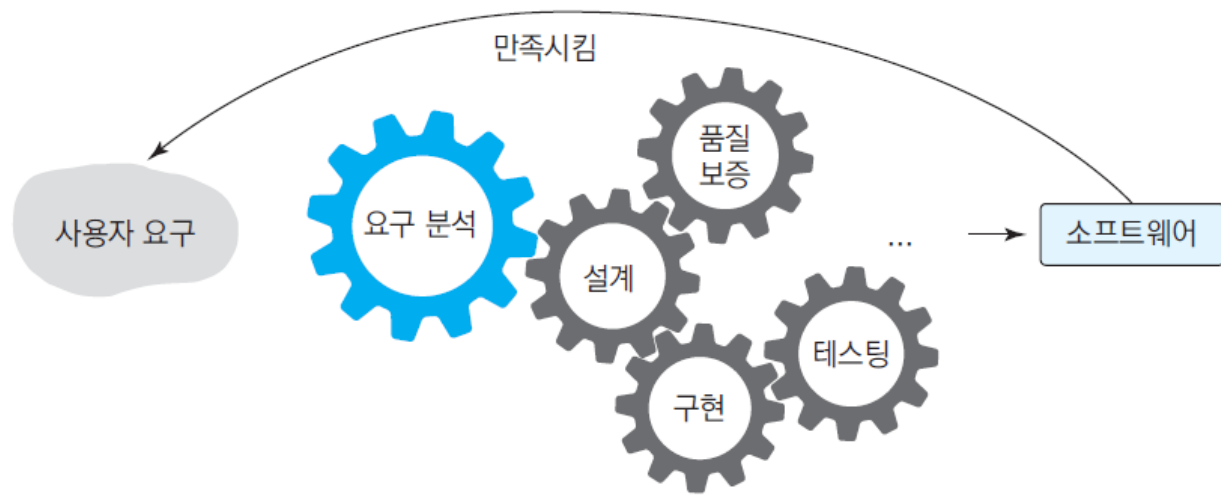


그림 1.7 공학적 접근법

소프트웨어 공학의 목표

- 복잡도 낮춤
- 비용 최소화
- 개발 기간 단축
- 대규모 프로젝트 관리
- 고품질 소프트웨어
- 효율성



품질



생산성

- 여러 가지 '원리와 방법'을 적용하여
 - 품질 좋은 소프트웨어를
 - 최소의 비용으로
 - 계획된 일정에 맞추어 개발하는 것

소프트웨어 공학의 주제

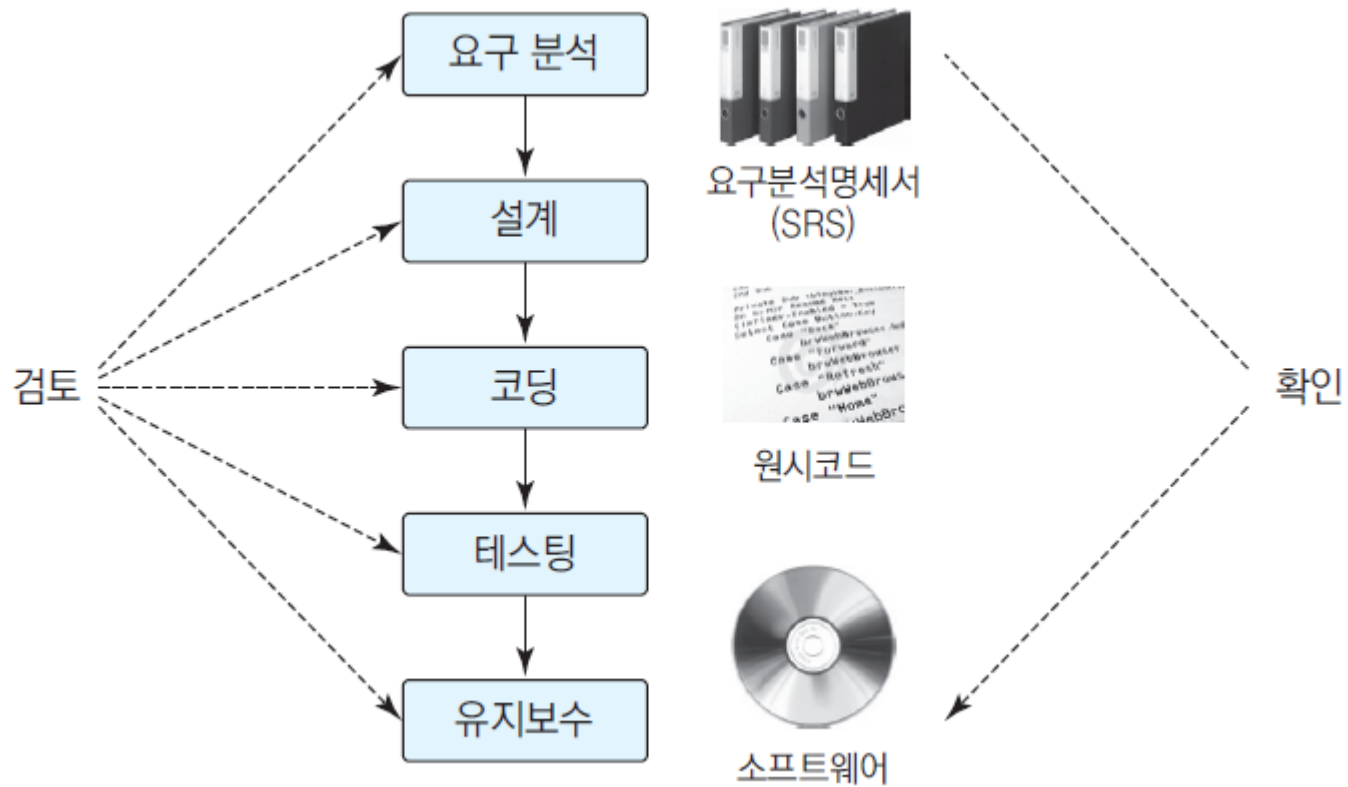
- 단계적 프로세스
- 품질 보증
- 프로젝트 관리

단계적 프로세스

- 코딩에 치중하지 않고 요구분석, 설계, 코딩, 테스트 등 정해진 절차를 따라 작업하는 것



품질보증



프로젝트 관리

- 프로젝트 계획
- 자원 관리
- 리스크 관리
- 프로젝트 수행과 모니터링

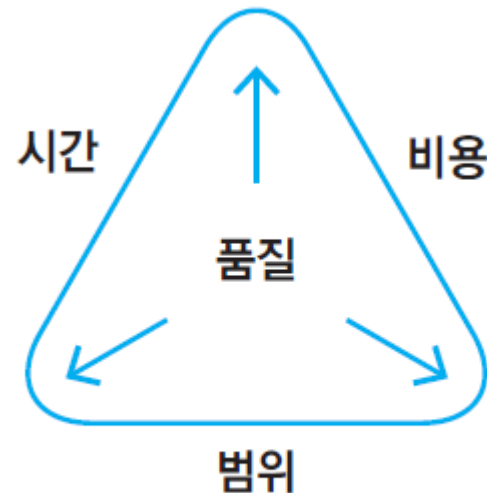


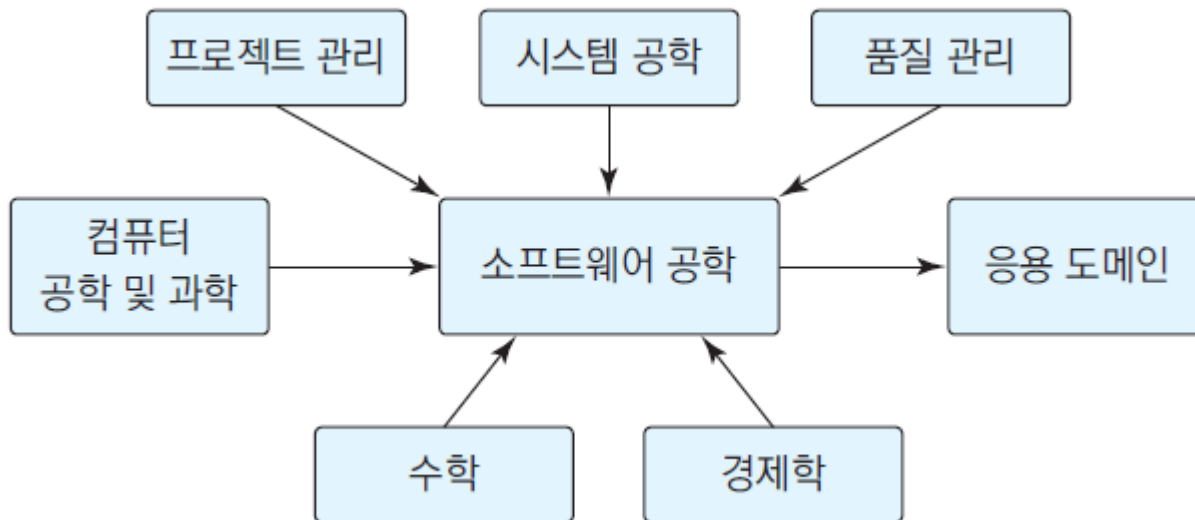
그림 1.11 프로젝트의 세 가지 제약

소프트웨어 공학의 연구 결과

주제	의미	사례	요리에 비유
방법 (method)	소프트웨어 제작에 사용하는 기법이나 절차	<ul style="list-style-type: none">• 구조적 분석, 설계 방법• 객체지향 분석, 설계 방법	익히는 방법 (구이, 찜, 훈제 등)
도구 (tool)	자동화된 시스템	<ul style="list-style-type: none">• 설계 도구• 프로그래밍 도구• 테스트 도구	요리 도구 (프라이팬, 압력 솥, 오븐 등)
프로세스 (process)	도구와 기법을 사용하여 작업하는 순서	<ul style="list-style-type: none">• Unified Process• eXtreme Programming	조리 순서(레시피)
패러다임 (paradigm)	접근 방법, 스타일	<ul style="list-style-type: none">• 구조적 방법론• 객체지향 방법론	음식 스타일 (한식, 일식, 중식, 퓨전 등)

1.5 연관 분야

- 두 가지 분야
 - 컴퓨터 공학의 원리나 기술과 관련된 여러 원리
 - 이를 적용하여 특정한 문제를 해결하려는 응용 도메인



컴퓨터 과학과 소프트웨어 공학

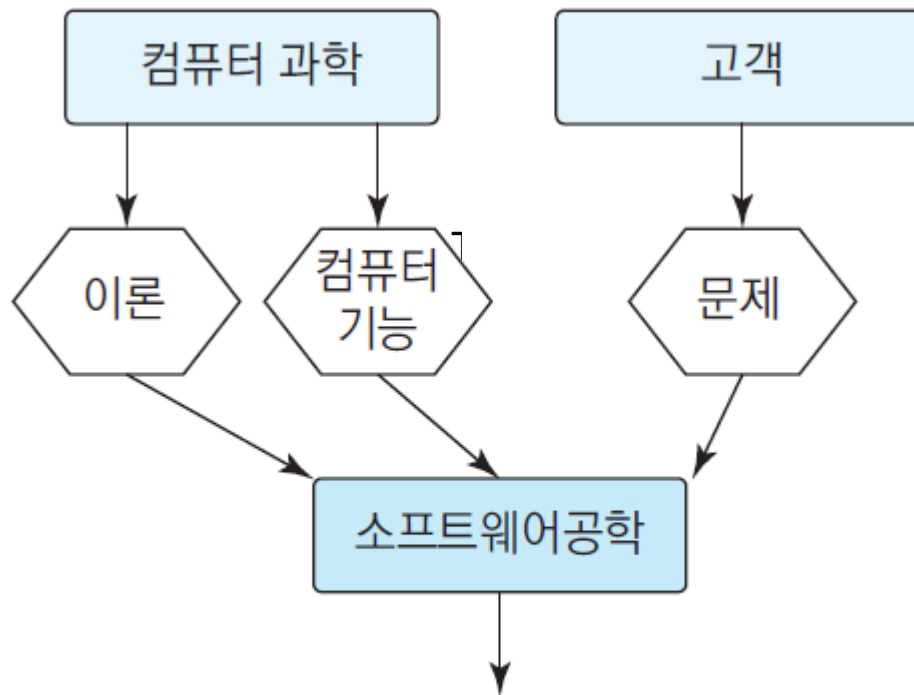


그림 1.14 소프트웨어 공학과 컴퓨터 과학