쉽게 배우는

소프트웨어 공학

2판

[강의교안 이용 안내]

IT@COOKBOOK

쉽게 배우는

소프트웨어 공학

2판

Chapter 04 요구분석

목차

학습목표

- 요구분석에 대해 살펴본다.
- 요구사항 표현 방법을 알아본다.
- 유스케이스 다이어그램 작성 방법을 자세히 배운다.

01. 요구사항

■ 요구사항

▪ 요구사항의 정의

- 사전적 정의: '이용자가 어떤 문제를 풀거나 목표를 달성하는 데 필요한 조건이나 능력'
- 소프트웨어 개발에서의 정의: '사용자와 개발자가 합의한 범위 내에서 사용자가 필요로 하는 기능'
  시스템이 제공하는 기능 요구와 품질과 같은 비기능 요구로 나뉨
- 요구사항이 정확히 무엇인지 파악하는 작업은 요구분석 단계에서 이루어짐

그림 4-1 사용자의 요구사항을 파악하는 생산자

02. 요구분석의 이해

■ 요구분석의 정의와 목적

- 집짓는 과정의 예시
  - 집을 지으려는 사람은 본인이 수집한 자료와 구상한 내용을 토대로 건축 설계사에게 어떤 집을 짓고 싶은지 설명
  - 본인이 가지고 있는 예산과 언제 입주하고 싶은지, 건축에 필요한 제반 사항 등에 대해서도 이야기를 나눔
  - 이 과정에서 고객은 어떤 건물을 원하는지 충분히 설명할 수 있고 건축 설계사는 고객이 원하는 것을 설계 도면에 빠짐없이 반영할 수 있음

그림 4-2 건축 설계 과정

02. 요구분석의 이해

■ 요구분석의 정의와 목적

- 요구분석의 정의
  - 컴퓨터 용어사전: '시스템이나 소프트웨어의 요구사항을 정의하기 위해 사용자 요구사항을 조사하고 확인하는 과정
  - 요구분석은 소프트웨어 개발 성패의 열쇠
- 요구분석의 목적
  - 목적사용자에게서 필요한 요구사항을 추출해 목표하는 시스템의 모델을 만들고 '요구분석명세서'를 작성하기 위해서
  - 요구분석명세서
    - 요구분석 단계에서 생성 되는 최종 산출물로 시스템의 기능이 무엇인지(what)에만 초점을 두고 정리
    - 요구분석단계 후 설계 단계에서는 '설계서'가 만들어지는데 이 문서는 어떻게(how) 구현할지 기술

그림 4-3 '무엇'과 '어떻게'

02. 요구분석의 이해

■ 요구분석의 정의와 목적

- 요구분석 관련자(대학 학사관리시스템 개발의 예시)
  - 발주사: 외주 방식으로 학사관리시스템 개발을 의뢰한 A대학교
  - 경영자(총 책임자): 학사관리시스템 개발을 결정한 A대학교 총장
  - 발주 담당자: 학사관리시스템 개발을 외주로 진행하는 모든 절차를 준비하는 담당자
  - 사용자: 외주 방식으로 개발된 학사관리시스템을 실제 사용하는 사람 (학생, 교수, 조교, 학사담당직원 등)
  - 수주사: A대학교의 학사관리시스템 개발을 위한 응찰 경쟁에서 이겨 개발이 결정된 업체 (B개발사)
  - 분석가: 사용자 요구사항이 무엇인지 파악하고 추출 및 정리하는 것까지 담당
    B개발사의 분석가는 A대학교를 방문해 현행 시스템 의 현황과 문제점을 파악하고 새로운 요구사항을 수집해 최종 산출물인 요구분석명세서를 작성
  - 설계자: 요구분석명세서를 바탕으로 아키텍처 설계, 모듈 설계, 데이터베이스DB 설계, 사용자 인터페이스 설계 등을 담당
  - 개발자: 넓은 의미에서 개발자는 학사관리시스템 개발에 참여하는 B개발사의 분석가, 설계자, 프로그래머 등을 포함
    좁은 의미에서는 프로그래머

02. 요구분석의 이해

■ 요구분석의 어려움

- 문제 영역에 대한 분석가의 이해력 부족
  - 법적인 문제를 해결할 경우 해당 분야에 전문적인 변호사를 선임
  - 요구분석가는 IP 분야의 전문가라 문제 영역(회계 분야, 금융 분야, 기업의 통합관리 등)에 대한 이해력이 부족할 수 있음
  - 사용자 요구를 잘못 이해한 채 분석해 필수 요구사항을 빠뜨릴 수도 있음
  - 이런 문제를 최소화하려면 문제 영역과 관련된 프로젝트를 개발한 경험이 많은 분석가를 투입하는 것이 바람직

그림 4-4 분야별 전문 변호사의 활동 모습

그림 4-5 경험 많은 분석가들

02. 요구분석의 이해

■ 요구분석의 어려움

- 사용자와 분석가의 의사소통 문제
  - 소프트웨어 개발에서는 견본이 없는 경우가 대부분이므로 사용자가 분석가에게 요구 사항을 설명하기가 어려움
  - 원하는 바를 분석가에게 어떻게 설명해야 할지 잘 모름
  - 필요한 요구사항은 반영하지 못하고 불필요한 사항만 포함되기도 함
  - 의사소통 문제는 개발자 간에도 있을 수 있음
- 사용자의 계속되는 요구사항 추가
  - 사용자의 처음 요구는 단순하고 간단할 수 있지만 분석가와 대화 후 점점 새로운 생각이 늘어나거나 관련 지식이 늘어남
  - 그에 따라 새로운 요구가 생기고 요구사항이 계속 추가될 수 있음
  - 소프트웨어 개발 과정 중에 목표 환경이 달라지면서 요구사항이 변경되기도 함

그림 4-6 요구사항을 계속 추가하는 사용자

02. 요구분석의 이해

■ 요구분석의 어려움

- 사용자의 모호한 요구사항
  - 사용자는 분석가에게 모호하게 요구사항을 전달하면 분석가가 해석할 수 있는 여지가 많으므로 오해로 인한 문제가 발생
  - 세부적인 내용을 전달하지 않는다면 분석가는 세부적인 요구사항을 본인의 경험에 비추어 해석해 사용자의 의도와 달라짐
  - 부서 간의 요구가 서로 어긋나고 경영진, 실무자의 요구가 상반되어 일관성이 없고 모순되는 요구사항이 발견될 수 있음
  - 분석가는 이러한 요구를 정리해서 반영 하기 위해 이해 당사자 간의 주장을 조율해야 함

그림 4-7 조율에 능숙해야 하는 분석가

02. 요구분석의 이해

■ 요구사항 수집

- 자료 수집
  - 가장 먼저 할 일은 A대학교의 현행 시스템에서 모든 업무의 입력 화면과 출력물을 받아 기본 기능을 분석하는 것
  - 관련 직원으로부터 현행 시스템의 문제점과 개선점, 새로 추가되어야 할 요구사항을 파악
- 인터뷰
  - 가능하면 먼저 자료를 수집한 후 이를 정리해 분석한 결과를 바 탕으로 각 부서 담당자를 인터뷰하는 것이 효율적
  - 미리 받은 자료를 토대로 대화를 해야 큰 줄기를 흩트리지 않고 계획대로 진행할 수 있음
  - 오랜 시간 동안 대화하기보다는 핵심 요구사항을 빨리 습득하는 것도 능력
- 설문 조사
  - 설문 조사를 통해 또 한 번의 요구사항을 도출할 수 있음
  - 중요한 것은 효율적인 설문 조사를 위해 설문 문항을 잘 만들어야 한다는 것

03. 요구분석 절차와 요구사항 종류

■ 요구분석 절차와 요구사항 분류

▪ 요구 분석 절차

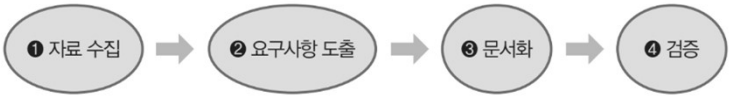

그림 4-8 요구분석 절차

❶ 자료 수집

현행 시스템의 문제점 도출, 실무 담당자 인터뷰, 현재 사용하는 문서 검토 등을 통해 가능한 모든 자료를 수집

❷ 요구사항 도출

수집한 자료를 정리해 적절히 분류하고 개발에 반영할 요구사항을 도출

❸ 문서화

도출한 요구사항을 요구분석명세서로 작성

❹ 검증

요구분석명세서에 사용자 요구가 정확히 기록되어 서로 모순되는 사항은 없는지, 빠뜨리지 않고 전부 기록했는지 등을 점검

03. 요구분석 절차와 요구사항 종류

■ 요구분석 절차와 요구사항 분류

▪ 요구 분석 분류

- 사용자 요구사항은 개발될 소프트웨어가 제공할 기능 요구사항과 성능, 제약 사항, 품질과 같은 비기능 요구사항으로 분류

그림 4-9 요구사항 분류

03. 요구분석 절차와 요구사항 종류

■ 기능 요구사항

- 사용자가 원하는 기능
- 사용자는 시스템을 통해 기능을 제공받기 바라며 시스템은 사용자에게 필요한 기능을 제공
- 사용자가 원하는 기능은 요구분석명세서에 완전하고 일관성 있게 표현해야 하며 시스템에도 전부 반영
- 완전성: 사용자가 원하는 모든 기능이 포함 / 일관성: 요구사항 간에 모순이 있어서는 안 됨

■ 비기능 요구사항

- 비기능 요구사항의 개요
  - 수행 가능한 환경, 품질, 제약 사항 등을 말함
  - 비기능 요구사항이 매우 중요한 소프트웨어에는 군사 무기나 의료 장비 등이 해당

▪ 제약 사항

- 개발 소프트웨어가 수행될 환경에 의한 조건
- 예시
  - 자바 언어를 사용해 개발하고, CBD 개발 방법론을 적용해야 함
  - 레드햇 리눅스 엔터프라이즈 버전에서 실행해야 함
  - 웹 로직 서버를 미들웨어로 사용해야 함
  - 윈도 운영체제와 리눅스 운영체제에서 모두 실행할 수 있어야 함

03. 요구분석 절차와 요구사항 종류

■ 비기능 요구사항

▪ 기능

- 신뢰성
  - 소프트웨어에서의 신뢰성: 소프트웨어를 믿고 사용할 수 있는 것
  - 일반적으로 신뢰성은 신뢰도로 나타내며 신뢰도는 '장애 없이 동작하는 시간의 비율'로 나타냄
  - 소프트웨어를 1,000번 수행했을 때 오류 없이 동작하는 횟수가 999번인 경우: 99.9%
  - 1,000시간 작동하는 동안 1시간의 장애가 있었던 경우: 99.9%(신뢰도 측정은 가용성(이용 가능성)을 척도로 사용)

- 신뢰도 높은 소프트웨어의 조건
  - 시스템에 오류가 있더라도 고장을 일으키지 않고 회피할 수 있는 능력이 높아야 함
  - 고장이 발생하더라도 이전 수준으로 성능이 회복되어야 함
  - 고장으로 영향을 받은 데이터들이 정상적으로 잘 복구됨

03. 요구분석 절차와 요구사항 종류

■ 비기능 요구사항

- 기능
  - 가용성
    - 소프트웨어가 총 운용 시간 동안 얼마나 정상적으로 고장 없이 가동되었는지를 비율로 나타냄

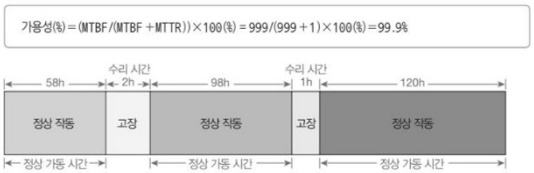

그림 4-10 신뢰도 계산

    - MTBF(평균 정상 작동 시간 또는 고장 간 평균 시간)
      - 고장이 나서 수리 후 정상 작동 하다가 다시 고장이 날 때까지 작동한 시간
      - 정상 작동 시간의 합을 횟수로 나눈 값
      - MTBF와 비교되는 개념으로 고장까지의 평균 시간 (고장은 수리가 불가능한 상황에 해당)
    - MTTR(평균 수리 시간)
      - 수리 시간의 합을 횟수로 나눈 값
      - 수리 시간: 고장 시점에서 수리가 완료되어 재가동되는 시점까지의 시간

04. 요구사항의 표현

■ 표현과 모델의 이해

- 표현
  - 생각하거나 느낀 것을 표현하는 방법은 다양
  - 수학 공식은 글로 표현하면 너무 길고 불분명해지는 내용을 공통으로 이해할 수 있고 오해의 소지가 없는 특정 기호로 표현
- 모델
  - 표현을 위해 사용하는 악보나 수학 공식은 하나의 모델
  - 장난감 자동차나 로봇, 생명 과학자들이 사용하는 DNA 분자 모형 등 또한 모델의 종류

그림 4-11 다양한 표현 방법

04. 요구사항의 표현

■ 모델의 정의와 필요성

- 모델의 정의와 필요성
  - '복잡한 대상의 핵심 특징만 선별해 일정한 관점으로 단순화한 뒤 기호나 그림 등을 사용해 체계적으로 표현한 것
  - 모델이 필요한 가장 큰 이유는 실제 모습을 미리 확인하기 위함(예: 모델하우스)
  - 하나의 사물도 여러 모델이 필요(예: 건물 건설시 여러 관점과 사용 목적에 따라 다수의 도면이 필요)
- 소프트웨어 개발 모델
  - 건축에 여러 도면이 필요하듯이 소프트웨어를 개발할 때도 여러 종류의 모델이 필요
  - 객체지향 개발에서는 UML의 다양한 다이어그램을 통해 개발 소프트웨어의 범위나 개략적인 구조와 기능을 이해할 수 있음
  - UML의 수많은 다이어그램이 소프트웨어 개발 과정에서 하나의 모델로 사용

04. 요구사항의 표현

■ 모델의 정의와 필요성

- 소프트웨어 개발 모델
  - 소프트웨어 개발 모델 사용의 장점
    - 이해도 향상
      - 모델을 사용하려면 실제 개발 범위에 해당하는 복잡한 내용을 단순화해야 하므로 개발 영역을 더 정확히 이해할 수 있음
      - 프로그램의 전체 모습을 시각적으로 나타낼 수 있으므로 사용자는 관련 요소들의 상호 관계를 빨리 파악할 수 있음
      - 이해 당사자 간의 의사소통 도구로 활용할 수 있음
    - 유지보수 용이
      - 추후 요구사항 변경에 따른 유지보수에 활용할 수 있음
      - 이 후 유사한 시스템을 개발할 때도 활용해 개발 기간과 비용을 줄여줌
  - 소프트웨어 개발 모델 사용의 장점
    - 과도한 문서 작업으로 일정 지연
      - 단계마다 각종 다이어그램을 포함한 문서 작성에 많은 시간을 할애하다 보면 실제 개발 업무가 지연될 수 있음
    - 형식적인 산출물로 전락할 가능성
      - 단계마다 수많은 문서가 형식적으로만 만들어져 활용되지 못할 수 있음
      - 변경된 내용을 산출물에 즉시 반영하지 않으면 문서 자체가 사장될 가능성이 높음

04. 요구사항의 표현

■ 모델링

- 모델링의 개요
  - 앞서 설명한 모델을 제작하는 과정 또는 작업
  - 연필로 종이에 간단한 콘티를 작성하거나 스토리보드같이 자기가 표현하고 싶은 것을 문장으로 서술하는 것도 한 종류
  - 각 전문 분야의 모델링은 기호나 다이어그램, 표기법 등을 사용해 차이가 없이 일관성 있는 해석이 가능하도록 표현

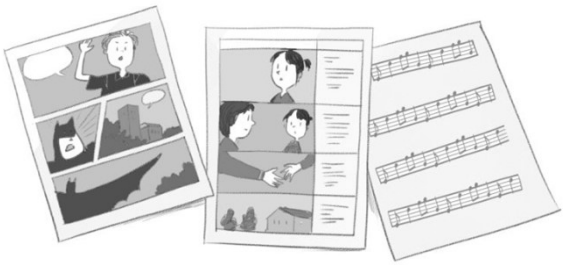

그림 4-12 다양한 형태의 모델링

04. 요구사항의 표현

■ 모델링

- 자연어를 사용한 표현
  - 사용자 요구사항을 자연어로 표현(모델링)하는 것
  - 도구나 기술을 따로 익힐 필요가 없 고 사용자와 대화할 때도 이해하기 쉬움
  - 표현이 길어질 수 있고 해석이 달라질 수 있으며 표현된 내용을 검증하기가 어려움
- 형식 언어를 사용한 표현
  - 친숙하지는 않지만 문법과 의미가 수학을 기초로 작성되어서 간결하고 정확하게 표현할 수 있음
  - 표기법을 별도로 공부해야 하므로 일반 개발자가 사용하기에는 조금 어려운 편
    - 관계형 표기법: 함의 방정식, 순환 관계, 대수 공리, 정규 표현법을 사용
    - 상태 위주 표기법: 의사결정 테이블, 이벤트 테이블 , 전이 테이블, 유한 상태 기계, 페트리 넷을 사용
- UML 다이어그램을 사용한 표현
  - UML 다이어그램을 사용하면 개발할 소프트웨어를 가시적으로 볼 수 있고, 개발할 소프트 웨어에 대한 문서화도 가능
  - 이 문서들은 분석 및 설계 과정에서 유용하며 검증 자료로도 활용

04. 요구사항의 표현

■ 모델링 언어

▪ 모델링 언어의 정의

- 모호한 표현의 문제를 해결하기 위해 사용하는 기호, 표기법, 도구
- 악보 기호, 수학 기호 등과 UML 다이어그램, Z 언어와 같은 형식적 표기법 등

• 소프트웨어 개발에서의 모델링 언어

- 요구사항 정의 및 분석·설계의 결과물을 다양한 다이어그램으로 표현하는 표기법
- 모호한 표현이 없 고 일관되어 모델링을 하는 데 매우 유용
- 개발자 간에 원활하게 의사소통할 수 있고 시스템을 주관적으로 해석하는 일도 줄어듬
- 다양한 다이어그램을 사용하므로 시스템을 여러 시각으로 볼 수 있음
- 사용자 요구사항을 검증하는 의사소통 도구로도 사용

• 개발 방법에 따른 모델링 언어

- 구조적 방법: 자료 흐름도(DFD), 자료 사전(DD), 소단위명세서를 사용해 요구분석 결과를 표현
- 정보공학 방법: E-R 다이어그램(ERD)을 데이터베이스 설계 표현을 사용
- 객체지향 방법: 이 방법은 대부분의 나라에서 공통으로 사용하는 UML 표기법을 사용
  UML 표기법은 10여 개의 다이어그램으로 구성되어 있으며 특히 요구사항은 유스케이스 다이어그램을 사용해 표현

04. 요구사항의 표현

■ 구조적 방법의 표현 도구: DFD

▪ 자료 흐름도

- 자료 흐름도: 들어온 자료를 처리한 후 결과를 내보내는 과정을 나타낸 것

• 자료 흐름도 작성 단계

- [1단계] 처리 표현: DFD를 작성할 때는 먼저 필요한 기능이 무엇인지 찾아내야 함
- [2단계] 입력과 출력 자료 표현: 학점계산에 대한 입력 자료와 출력 자료를 찾아 화살표 위에 나타냄
- [3단계] 자료 저장소 표현: 처리할 자료가 많아 데이터베이스 테이블로 저장되어 있다면 자료 저장소로 나타냄
- [4단계] 자료 출원지와 목적지 표현: 자료의 출원지와 목적지도 DFD에 표현해야 함
  자료의 출원지와 목적지는 사각형 박스로 표현 (터미네이터)

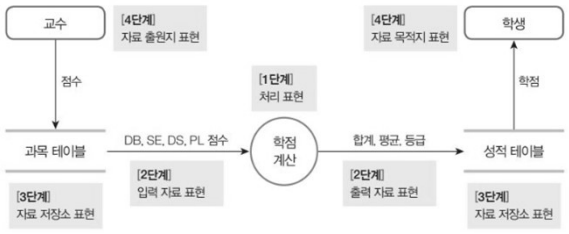

그림 4-13 자료 흐름도 작성의 예

04. 요구사항의 표현

■ 구조적 방법의 표현 도구: DFD

- 자료 사전
  - 자료 사전(DD): 자료 흐름도(DFD)에 존재하는 데이터에 대한 자세한 설명을 기록해 놓은 것
  - 자료 사전은 몇 가지 기호를 사용해 자료를 정의

표 4-1 자료 사전의 기호

기호	의미	설명
=	정의is composed of	구성 내용을 설명하는 것으로 '~로 구성되어 있다'로 해석한다.
+	연결and	'~와'로 해석한다.
()	생략 가능optional	생략할 수 있다.
[|]	선택choose only one of	|로 분리된 둘 중 하나만 선택할 수 있다.
{ }	반복iteration of	여러 개 존재함을 나타내며 '{' 왼쪽에 숫자를 넣으면 최소 반복횟수, '}' 오른쪽에 숫자를 넣으면 최대 반복횟수가 된다.
**	주석comment	부연 설명을 나타낸다.

04. 요구사항의 표현

■ 구조적 방법의 표현 도구: DFD

- 자료 사전
  - 자료 사전 작성의 예

```
교과목=과목번호+과목명+이수 구분+학점("시수")+개설 학년("개설 학기")
과목번호=*교과목을 구별할 수 있는 유일한 키 값으로 사용된다.*
학점=[1|2|3]
시수=0| 과목에 대한 주당 시수를 말한다.
이수 구분=[대기|전필|전선|교직]
        =*별칭: 대기=대학 기초, 전필=전공 필수, 전선=전공 선택*
개설 학년=[1|2|3|4]
개설 학기=[1|2|계절학기]
```

그림 4-14 자료 사전 작성의 예

- 교과목은 과목번호, 과목명, 이수 구분, 학점(시수), 개설 학년(학기)으로 구성
- 과목번호는 교과목을 구별할 수 있는 유일한 키 값으로 사용
- 학점은 1학점 또는 2학점 또는 3학점 중 하나
- 시수는 '1주일에 몇 시간 강의하는지'를 나타냄
- 이수 구분은 대학 기초 또는 전공 필수 또는 전공 선택 또는 교직 중 하나
- 개설 학년은 1학년 또는 2학년 또는 3학년 또는 4학년 중 하나
- 개설 학기는 1학기 또는 2학기 또는 계절학기 중 하나

04. 요구사항의 표현

■ 구조적 방법의 표현 도구: DFD

▪ 소단위명세서

- 소단위명세서: DFD에서 원으로 표시한 프로세스인 처리는 실제 알고리즘 형태로 작성한 것
- 소단위명세서를 작성하는 도구에는 구조적 언어, 선후 조건문, 의사 결정표 등이 있음

```
int DB, SE, DS, PL;
scanf(DB, SE, DS, PL)
print(합계 : DB + SE + DS + PL)
print(평균 : DB + SE + DS + PL)/4)
print(등급 : grade(DB), grade(SE), grade(DS), grade(PL))

grade(score) {
   if (score >= 90) {return 'A'}
   else if (score >= 80) {return 'B'}
   else if (score >= 70) {return 'C'}
   else if (score >= 60) {return 'D'}
   else {return 'F'}
```

그림 4-15 소단위명세서 작성의 예

04. 요구사항의 표현

■ 정보공학 방법의 표현 도구: E-R 다이어그램

▪ E-R 다이어그램

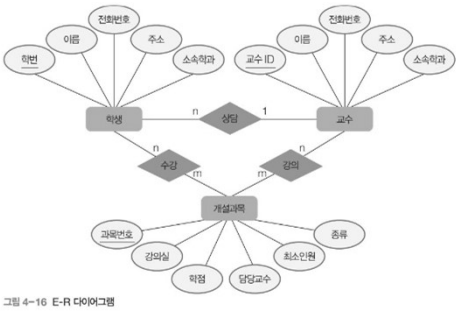

그림 4-16 E-R 다이어그램

- 교수는 여러 명의 학생을 상담
- 여러 명의 학생은 여러 과목을 수강
- 교수 ID는 주 키로 사용
- 학번은 주키로 사용
- 과목번호는 주 키로 사용
- 교수는 여러 과목을 강의
- 교수 객체는 교수 ID, 이름, 전화번호, 주소, 소속학과를 속성으로 가짐
- 학생은 학번, 이름, 전화번호, 주소, 소속학과를 속성으로 가짐
- 개설과목은 과목번호, 강의실, 학점, 담당교수, 최소인원, 종류를 속성으로 가짐

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

▪ 유스케이스 다이어그램 작성하기

• [1단계] 후보 유스케이스 도출

표 4-2 학사관리시스템의 1차 유스케이스 목록

사용자	필요한 기능
교수	① 교과목(등록/수정/삭제/조회) ② 개설과목(등록/수정/삭제/조회) ③ 개설과목 수강인원(등록/수정/삭제/조회) ④ 성적(입력/수정/삭제/조회) ⑤ 강의계획서(입력/수정/삭제/조회) ⑥ 출석부(등록/수정/삭제/조회) ⑦ 담당 과목 수강 현황 조회
학생	① 수강과목(신청/수정/삭제/조회) ② 학생정보(등록/수정/조회/삭제) ③ 조회(교과목, 개설과목, 성적, 강의계획서, 장학생 명단)
조교	① 교과목(등록/수정/삭제/조회) ※ 대행(등록/수정/삭제) ② 개설과목(등록/수정/삭제/조회) ※ 대행(등록/수정/삭제) ③ 장학생 명단(등록/수정/삭제/조회) ④ 개설과목 수강인원(등록/수정/삭제/조회)
학사담당직원	① 기간 설정(교과목등록, 개설과목등록, 수강신청, 수강신청변경)

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

▪ 유스케이스 다이어그램 작성하기

• [2단계] 후보 유스케이스 검토 / 교수 액터가 사용하는 유스케이스

1. 교과목(등록/수정/삭제/조회): 모든 기능 사용
2. 개설과목(등록/수정/삭제/조회): 모든 기능 사용
3. 개설과목 수강인원(등록/수정/삭제/조회): 후보 유스케이스에서 탈락
4. 성적(입력/수정/삭제/조회): '삭제' 기능 제거
5. 강의계획서(입력/수정/삭제/조회): '삭제' 기능 제거
6. 출석부(등록/수정/삭제/조회): → '등록/수정/삭제' 기능 제거
7. 담당 과목 수강 현황 조회: 후보 유스케이스에서 탈락

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 다이어그램 작성하기
  - [2단계] 후보 유스케이스 검토 / 학생 액터가 사용하는 유스케이스
    1. 수강과목(신청/수정/삭제/조회): 수정' 기능 제거
    2. 수강과목(신청/수정/삭제/조회): '등록'/'삭제' 기능 제거
    3. 조회(교과목, 개설과목, 성적, 강의계획서, 장학생명단): 모든 기능 사용
  - [2단계] 후보 유스케이스 검토 / 조교 액터가 사용하는 유스케이스
    1. 교과목(등록/수정/삭제/조회): 모든 기능 사용
    2. 개설과목(등록/수정/삭제/조회): 모든 기능 사용
    3. 장학생 명단(등록/수정/삭제/조회): 모든 기능 사용
    4. 개설과목 수강인원(등록/수정/삭제/조회): 이미 후보 유스케이스에서 탈락
  - [2단계] 후보 유스케이스 검토 / 학사담당직원 액터가 사용하는 유스케이스
    1. 기간 설정(수강신청, 수강신청변경, 교과목등록, 개설과목등록): 모든 기능 사용

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 다이어그램 작성하기
  - [2단계] 후보 유스케이스 검토

표 4-3 1차 검토 후 유스케이스 목록

사용자	유스케이스 검토 현황	제거된 기능
교수	① 교과목(등록/수정/삭제/조회)	없음
	② 개설과목(등록/수정/삭제/조회)	없음
	③ 개설과목 수강인원(등록/수정/삭제/조회)	유스케이스 탈락
	④ 성적입력/수정/조회)	'삭제' 기능 제거
	⑤ 강의계획서(입력/수정/조회)	'삭제' 기능 제거
	⑥ 출석부(조회)	'등록/수정/삭제' 기능 제거
	⑦ 담당 과목 수강 현황 조회	유스케이스 탈락
학생	① 수강과목(신청/수정/삭제/조회)	'삭제' 기능 제거
	② 학생정보(등록/수정/조회/삭제)	'등록/삭제' 기능 제거
	③ 조회(교과목, 개설과목, 성적, 강의계획서, 장학생 명단)	없음
조교	① 교과목(등록/수정/삭제/조회)	없음
	② 개설과목(등록/수정/삭제/조회)	없음
	③ 장학생 명단(등록/수정/삭제/조회)	없음
	④ 개설과목 수강인원(등록/수정/삭제/조회)	유스케이스 탈락
학사담당직원	① 기간 설정(교과목등록, 개설과목등록, 수강신청, 수강신청변경)	없음

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 다이어그램 작성하기
  - [2단계] 후보 유스케이스 검토 / 유스케이스 특성 확인 및 통합
    - 실행 후 사용자가 원하는 결과 제공: 결과물을 제공하는 지 다시 검토
    - 사용자에 의해 시작: 어떤 사용자가 사용하는지, 어 떤 사용자가 시작하는지를 생각
    - 많이 사용하는 기능은 하나의 유스케이스로 도출: '등록/수정/조회/삭제' 기능은 하나의 유스케이스로 만드는 것이 타당
    - 사용하는 기능만 유스케이스로 도출
      - 여러 기능을 단순화해 하나의 유스케이스로 만들면 사용자는 그 안에 포함된 기능을 모두 사용해야 함

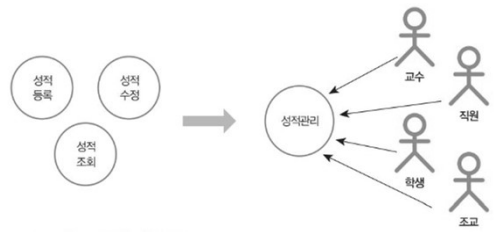

그림 4-17 기능을 잘못 묶은 유스케이스

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 다이어그램 작성하기
  - [2단계] 후보 유스케이스 검토 / 유스케이스 특성 확인 및 통합
    - 주 기능은 유스케이스, 보조 기능은 이벤트 흐름
      - 가장 핵심이 되는 기능은 유스케이스로 나타내고 나머지는 이벤트 흐름에 나타냄

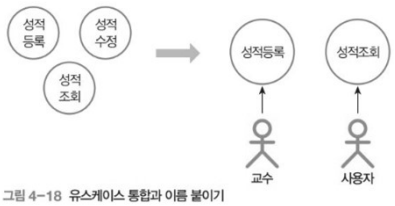

그림 4-18 유스케이스 통합과 이름 붙이기

- 홀로 존재하는 유스케이스는 없음: 홀로 존재하는 유스케이스는 불필요한 것일 가능성이 높으므로 재검토

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 다이어그램 작성하기
  - [3단계] 유스케이스 정련

표 4-4 최종 유스케이스 목록

사용자	최종 유스케이스	기타
교수	교과목등록	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스 '추가': 유스케이스의 이벤트 흐름 '삭제': 유스케이스의 이벤트 흐름
	개설과목등록	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스 '추가': 유스케이스의 이벤트 흐름 '삭제': 유스케이스의 이벤트 흐름
	성적 입력	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스
	강의계획서 입력	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스
학생	수강신청	'조회': 별도의 유스케이스 '추가': 유스케이스의 이벤트 흐름 '삭제': 유스케이스의 이벤트 흐름
	개인정보 수정	'조회': 별도의 유스케이스
	휴학신청	
조교	교과목등록대행	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스 '추가': 유스케이스의 이벤트 흐름 '삭제': 유스케이스의 이벤트 흐름
	개설과목등록대행	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스 '추가': 유스케이스의 이벤트 흐름 '삭제': 유스케이스의 이벤트 흐름
	휴학승인	
학사업무직원	시간 설정	'수정': 유스케이스의 이벤트 흐름 '조회': 별도의 유스케이스
사용자(학생, 교수, 조교, 학사업무직원)	조회	'교과목등록': 유스케이스의 이벤트 흐름 '개설과목등록': 유스케이스의 이벤트 흐름 '수강신청등록': 유스케이스의 이벤트 흐름 '수강신청변경': 유스케이스의 이벤트 흐름
교직원(교수, 조교, 학사업무직원)	출석부 조회	유스케이스, 이벤트, 흐름, 시나리오, 개설과목조회, 강의계획서조회, 수강신청조회, 학생정보조회, 성적조회, 장학생 명단 조회

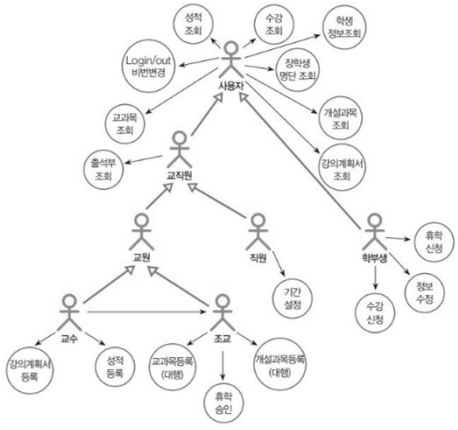

그림 4-19 학사관리 유스케이스 다이어그램

04. 요구사항의 표현

■ **객체지향 방법의 표현 도구: 유스케이스 다이어그램**

- 유스케이스 명세서 작성하기
  - 유스케이스 다이어그램을 작성하면 각각의 유스케이스에 대한 명세서를 작성해야 함
  - 유스케이스 명세서는 7개 항목으로 구성되며 각 유스케이스의 자세한 내용을 기술하는 문서

표 4-5 유스케이스 명세서 항목

항목	설명
개요	유스케이스가 제공하는 기능을 기술한다.
관련 액터	누가(역할) 이 유스케이스를 사용하는지 해당 액터를 기술한다.
우선순위	사용자 관점에서 중요하다고 생각하는 순서를 '상', '중', '하'로 기술한다.
선행 조건	유스케이스가 수행되기 전에 수행되어야 하는 조건을 기술한다.
후행 조건	유스케이스가 수행된 후 만족해야 하는 조건을 기술한다.
이벤트 흐름	유스케이스가 수행될 때 액터와 시스템의 대화(상호작용)를 기술한다.
비기능 요구사항	유스케이스와 관련 있는 비기능 요구사항을 기술한다.

04. 요구사항의 표현

■ 객체지향 방법의 표현 도구: 유스케이스 다이어그램

- 유스케이스 명세서 작성하기
  - 개요
    - 유스케이스가 제공하는 기능을 나타내는 것으로, 그 기능을 사용하는 사용자 액터까지 표현해야 함

유스케이스	개요
수강신청	학생은 시스템이 제공하는 개설과목 목록에서 수강할 과목을 선택한다. '과목삭제'와 '과목추가' 기능이 포함되어 있다.

    - 개요는 유스케이스에 포함된 모든 기능을 액터까지 포함해서 표현해야 함

유스케이스	개요
성적등록	교수는 학생의 성적을 입력한다. (×) 학생의 성적을 입력하고 수정한다. (×) 교수는 학생의 성적을 입력하고 수정한다. (O)

    - 유스케이스의 기능은 이벤트 흐름에서 별도로 자세히 표현하기 때문에 개요에서는 간략하게 나타냄

  - 관련 액터
    - 유스케이스를 사용하는 액터를 나타냄
    - 관련 액터 항목에 나타낸 액터와 유스케이스 다이어그 램에서 연관 관계를 맺고 있는 액터는 일치해야 함

유스케이스	관련 액터
수강신청	학생

04. 요구사항의 표현

■ 객체지향 방법의 표현 도구: 유스케이스 다이어그램

- 우선순위
  - 개발할 때의 순서를 나타낸 것
  - 기준을 정하고 그에 따른 개발 순서를 정한다면 프로젝트 관리 측면에서 도움이 됨
  - 우선순위는 사용자 관점에서 정할 수도 있고, 개발자 관점에서 정할 수도 있음
  - 중요도로 정하는 사용자 관점 우선순위: 일정 조절, 자원 배분, 인력 투입 등에서 균형 있게 프로젝트 관리를 할 수 있음
  - 개발 난이도로 정하는 개발자 관점 우선순위: 개발 난이도를 기준으로 '상', '중', '하'를 결정

- 선행 조건
  - 해당 유스케이스를 수행하기 위해 미리 만족해야 하는 상태나 조건
  - 해당 유스케이스를 수행하기 위한 직전의 선행 조건 유스케이스를 기입

유스케이스	선행 조건
수강신청	개설과목등록

- 후행 조건
  - 유스케이스 수행 후 만족해야 하는 상태
  - 상태나 수치의 바뀐 모습을 명확하고 구체적으로 작성
  - 모호하게 기술하지 않고 객관적으로 판단할 수 있게 작성

유스케이스	후행 조건
수강신청	수강신청을 한 총 학점이 표시되어야 한다.

04. 요구사항의 표현

■ 객체지향 방법의 표현 도구: 유스케이스 다이어그램

- 이벤트 흐름
  - 유스케이스 기능에 대한 액터와 시스템 간의 대화로 문장이나 순차 다이 그램을 사용해 표현
  - 사용자가 이해할 수 있는 쉬운 용어로 명확하게 작성해야 하며 당연히 개발자가 사용하는 기술적 용어는 사용해선 안됨
  - 작성된 이벤트 흐름은 개발자들이 설계와 테스트 단계에서도 활용
  - 유스케이스의 큰 줄기에 해당하는 정상적인 흐름은 기본 흐름으로, 나머지는 대안 흐름에 표현
  - 이벤트 흐름은 순차 다이어그램으로도 표현할 수 있음
  - 기본 흐름
    - 가장 일반적이고 정상적인 흐름
    - 액터와 시스템의 대화를 정리하다 보면 정상적인 흐름은 한 가지로 작성
  - 대안 흐름
    - 기본 흐름을 제외한 나머지 발생 가능한 여러 경우의 액터와 이벤트 간 대화 내용을 정리한 것
    - 예외적이거나 특수한 상황이 포함
- 비기능 요구사항 요구
  - 비기능 요구사항: 성능, 신뢰성, 보안성, 가용성 등 여러 요구 품질 요소를 나타냄
  - 요구분석명세서의 비기능 요구사항은 유스케이스와 직접 관련이 있는 비기능 요구사항을 기술하는 것

유스케이스	비기능 요구사항
도서 검색	3초 이내에 결과가 나타나야 한다. (×) 1백만 권까지 3초 이내에 결과가 나타나야 한다. (○)
수강신청	전교생이 동시에 접속해도 문제가 없어야 한다. (×) 수강신청 시 2,000명까지 동시 접속해도 1기가바이트 이상의 속도가 유지되어야 하고, 시스템이 멈추지 않아야 한다. (○)

04. 요구사항의 표현

■ 객체지향 방법의 표현 도구: 유스케이스 다이어그램

- 이벤트 흐름

학생	시스템
기본 흐름	
학생은 학사관리 메뉴에서 수강신청을 선택한다.	
	시스템은 학생정보를 확인해 미지막 학년/학기를 확인한다. 시스템은 수강신청할 학과/학년/학기의 개설과목을 디폴트로 보여준다. • 화면에 보이는 항목: 과목코드, 이수구분, 개설대학, 개설학과, 과목명, 학점/시간, 교수명, 대상 학년, 수강인원, 시간표, 강의실
학생은 희망하는 과목을 선택한 후 장바구니 버튼을 클릭한다.	
	시스템은 장바구니에 담긴 과목의 학점을 합산해 총 신청학점을 계산한다. 시스템은 장바구니에 담긴 과목리스트와 총 신청학점을 보여준다.
[신청할 과목이 더 없는 경우] 학생은 최종 버튼을 클릭한다.	
	시스템은 '수강신청이 완료되었습니다'라는 메시지 창을 띄워준다.
[신청할 과목이 더 있는 경우] 학생은 수강신청 메뉴에서 원하는 학과, 학년을 선택한다.	
	시스템은 해당 학과, 학년에 개설된 과목리스트를 화면에 보여준다.
학생은 원하는 과목을 선택한 후 장바구니 버튼을 클릭한다.	
	시스템은 장바구니에 담긴 과목리스트와 총 신청학점을 보여준다.
학생은 최종 버튼을 클릭한다.	
	시스템은 '수강신청이 완료되었습니다'라는 메시지 창을 띄워준다.

(계속)

대안 흐름: 취소	
학생은 수강신청 메뉴에서 취소를 선택한다.	
	시스템은 현재까지 수강신청된 과목리스트를 보여준다.
학생은 취소할 과목을 선택한다.	
	시스템은 취소 후 남거진 과목리스트를 보여준다.
학생은 최종 선택 버튼을 클릭한다.	
	시스템은 '수강신청이 완료되었습니다'라는 메시지 창을 띄워준다.
대안 흐름: 추가	
사용자는 수강신청 메뉴에서 추가를 선택한다.	
	시스템은 현재까지 수강신청된 과목리스트를 보여준다.
사용자는 원하는 학과, 학년을 선택한다.	
	시스템은 해당 학과, 학년에 개설된 과목리스트를 화면에 보여준다.
학생은 원하는 과목을 선택한 후 장바구니 버튼을 클릭한다.	
	시스템은 장바구니에 담긴 과목리스트와 총 신청학점을 보여준다.
학생은 최종 버튼을 클릭한다.	
	시스템은 '수강신청이 완료되었습니다'라는 메시지 창을 띄워준다.
대안 흐름: 초과 신청한 경우	
학생은 수강희망과목을 장바구니에 담고 최종 버튼을 클릭한다.	
	시스템은 '○학점을 초과했습니다'라는 메시지를 띄워주고 장바구니의 과목리스트를 보여준다.

그림 4-20 **수강신청 유스케이스에 대한 이벤트 흐름**

05. 요구사항의 문서화

■ 요구분석명세서의 이해

▪ 소프트웨어 요구분석명세서의 정의

- 요구분석 과정의 최종 산출물로 사용자와 개발자를 연결하는 중요한 문서
- 설계와 구현에서 참조할 사항, 전반적으로 알아야 할 사항을 포함하는 문서
- 사용자와 개발자의 계약서

• 요구분석명세서 작성

- 분석가와 사용자의 배경 지식이 너무 다르기 때문에 분석가가 요구분석명세서를 작성하는 일은 어려움
- 완전한 요구분석명세서를 작성하려면 분석가는 현행 업무 내용을 충분히 파악한 후 사용자와 대화할 수 있도록 준비
- 입출력 화면 등이 어떻게 구성되는지 사용자에게 설명하면서 사용자가 느끼는 문제점과 원하는 요구사항을 파악해야 함
- 분석가가 작성한 요구분석명세서는 사용자에게는 계약서, 개발자에게는 설계 및 구현을 위한 공식 문서로 사용
- 소프트웨어 개발과 관련된 이해 당사자에게는 중요한 기준이 되는 문서

05. 요구사항의 문서화

■ 요구분석명세서 작성 시 주의 사항

• 사용자가 쉽게 읽고 이해할 수 있도록 작성

- 사용자와의 관계에서 요구분석명세서는 계약서와 같은 효력이 있음
- 문제가 발생해 분쟁으로까지 이어질 때는 약관이 매우 중요한 근거 자료로 활용되고 잘잘못에 대한 판단 기준이 됨
- 상품의 약관처럼 작성해서는 안 되고 사용자가 쉽게 읽고 이해할 수 있도록 작성해야 함

그림 4-21 읽기도 어렵고 이해하기도 쉽지 않은 깨알 같은 글씨의 약관

05. 요구사항의 문서화

■ 요구분석명세서 작성 시 주의 사항

- 개발자가 설계와 코딩에 효과적으로 사용할 수 있도록 작성
  - 개발자가 해당 문서를 기반으로 설계하고 코딩하며 검증 기준으로 사용 하는 등 여러 목적으로 활용할 수 있게 제공해야 함
- 비기능 요구를 명확히 작성
  - 요구분석명세서에 서술된 기능과 제약 사항은 사용자와 개발자 모두 충분히 동의해야 함
  - 사용자는 적절한 개발비를 제시해야 하고 요구사항도 개발 기간 내에 끝낼 수 있을 정도여야 해야함
  - 개발자도 요구사항을 수용하기 위해 드는 비용과 기간, 불가능한 요구 등을 사용자에게 정확히 알려주고 이해시킴
  - 제약 사항과 같은 비기능 요구사항은 사용자가 명확히 알 수 있도록 작성
- 테스트 기준으로 사용할 수 있도록 정량적으로 작성
  - 소프트웨어 요구분석명세서는 테스트 기준을 작성하고 테스트 케이스를 만드는 데도 사용
  - 원하는 기능 요구사항뿐만 아니라 품질과 제약 사항 등의 비기능 요구사항을 가능한 정량적으로 명확히 작성

05. 요구사항의 문서화

■ 요구분석명세서 작성 시 주의 사항

- 품에 대한 우선순위 명시
  - 품질 특성이 서로 상충하는 경우에 어떤 특성을 우선순위에 둘 것인지를 명시해야 함
  - 상충 관계: 어느 한쪽이 좋으면 다른 한쪽은 손해를 보는 관계(예: 터널공사)
  - 요구사항의 품질도 이런 상충 관계가 성립될 수 있음(보안성 <-> 사용자 편의성)

그림 4-22 상충 관계의 예

05. 요구사항의 문서화

■ 잘 만든 요구분석명세서의 특성

- 완전성
  - '완전하다'라는 말은 빠진 부분 없이 모두 있다는 의미
  - 요구분석명세서는 기능 요구사항 뿐 아니라 성능, 제약 사항 등의 비기능 요구사항이 빠지지 않고 모두 서술되어야 완전
  - 요구사항 정의 및 분석 과정에서 아주 드물게 발생할 수 있는 것까지도 찾아내 반영해야 완전성을 충족하는 요구분석명세서
- 일관성
  - 상반된 요구뿐 아니라 불일치하거나 중복된 요구가 존재하면 일관성이 없는 요구분석명세서
  - 예) '아무나 접근할 수 없게 보안을 매우 철저히 해주고 사용자들이 인증 절차를 번거롭지 않고 단순하게 사용할 수 있음
- 명확성
  - 요구분석명세서도 계약서와 같은 역할을 수행하므로 요구사항, 비용, 기간 등에서 문제가 발생하면 근거가 됨
  - 요구사항을 잘못 해석해 의도하지 않은 결과를 초래할 수 있으므로 누구나 같은 해석을 하도록 표현을 명확히 해야 함

유스케이스	비기능 요구사항
도서 검색	3초 이내에 결과가 나타나야 한다. (×) 1백만 권까지 3초 이내에 결과가 나타나야 한다. (O)
수강신청	전교생이 동시에 접속해도 문제가 없어야 한다. (×) 수강신청 시 2,000명까지 동시 접속해도 1기가바이트 이상의 속도가 유지되어야 하고, 시스템이 멈추지 않아야 한다. (O)

05. 요구사항의 문서화

■ 잘 만든 요구분석명세서의 특성

- 추적 가능성
  - 추적이 가능하도록 요구분석명세서를 작성

그림 4-23 추적 가능성

05. 요구사항의 문서화

■ 잘 만든 요구분석명세서의 특성

- 변경 용이성
  - 한 곳의 변경으로 인해 다른 곳에 미치는 영향이 작은 것
  - 요구사항끼리 영향을 덜 미치려면 서로 의존도가 낮고 독립적으로 서술되어야 함
- 검증 가능성
  - 요구분석명세서가 시스템이 요구사항을 만족하는지를 체계적으로 검사할 수 있다면 검증 가능성은 큼
  - 요구분석명세서가 모호한 표현이 많이 포함되게 작성되었다면 검증 가능성은 작음
  - 예) '많은 학생이 동시에 수강신청을 해도 문제가 없어야 한다' 에서 '많은'은 주관적인 표현이므로 객관적인 수치로 변경

05. 요구사항의 문서화

■ 요구 명세 기법

- 비정형 명세 기법
  - 사용자 요구를 표현할 때 자연어를 기반으로 서술하거나 작업 흐름도와 같은 다이어그램을 사용해 작
  - 특별한 기술이 필요하지 않아 요구사항을 작성하기 쉬움
  - 명세 내용을 이해하기 쉬워 사용자와 분석가의 의사소통이 쉬움
  - 분석하는 과정에서 사용자의 적극적인 관심과 참여를 유도할 수 있음
  - 자연어로 작성하면 표현이 모호할 수 있고, 해석을 다르게 할 수도 있음
  - 일관성이 떨어질 수 있고 명세가 불충분할 수 있음
  - 작성된 내용이 사용자 요구를 충분히 반영해서 표현하고 있는지 완전성을 검증하기가 어려움
- 정형 명세 기법
  - 사용자의 요구를 수학적 원리와 표기법을 이용해 표현하는 것(Z 정형 명세 언어)
  - 사용자 요구를 정확하고 간결하게 표현할 수 있음
  - 수학에서 사용되는 증명 기술을 이용해 작성된 사용자 요구가 일관성이 있는지, 완전한지 등을 검증할 수 있는 장점이 있음
  - 사용자 요구를 정형화된 형태로 작성함으로써 효율적으로 테스트 케이스를 생성할 수 있음
  - 이 테스트 케이스는 명세 내용과 구현의 일치 여부를 확인하는 데 사용할 수 있음
  - 수학적 원리와 표기법을 사용하려면 분석가가 관련된 수학 지식이 있어야 함
  - 수학적 표현으로 작성된 표기법을 충분히 이해할 수 있어야 함
  - 이 표기법을 사용해 사용자 요구를 정확히 표현할 수 있어야 함

05. 요구사항의 문서화

■ 요구사항 검증

- 여러 방법을 통해 추출하고 정리한 사용자의 요구분석명세서가 정확하고 완전하게 서술되었는지 검토하는 활동
- 사용자 요구사항이 완전하게 서술되었는지 검증
- 요구분석명세서를 작성할 때 문서 표준을 따랐는지 확인
- 설계에서 사용하기에 적합한지를 확인

표 4-6 요구사항 검증 항목

검증 항목	설명
완전성completeness	사용자 요구사항이 빠짐없이 완전하게 반영되어 있는가?
일관성consistency	요구사항 간에 모순되거나 충돌되는 점은 없는가? 산출물 또는 요구사항의 내용이 일관적인가?
명확성unambiguity	서술된 명세서의 내용이 모호하지 않고 모든 참여자가 명확히 이해할 수 있는가?
기능성functionality	서술된 명세서의 내용이 '어떻게'보다 '무엇을'에 관점을 두고 서술되었는가?
검증 가능성verifiability	개발된 소프트웨어가 사용자가 요구하는 내용과 일치하는지를 검증할 수 있는가?
추적 가능성traceability	요구분석명세서, 설계서, 구현 코드에서 같은 내용을 쉽게 찾을 수 있는가?
변경 용이성modifiability	서로 의존 관계가 적어 변경으로 인해 미치는 영향이 적은가?

IT⊕C⊕⊕KBOOK

쉽게 배우는

소프트웨어 공학

2판

도서에 대한 의견을 남겨주세요.

의견 남기러 가기

감사합니다.