

02. 소프트웨어 개발 프로세스



■ 프로세스

- 프로세스

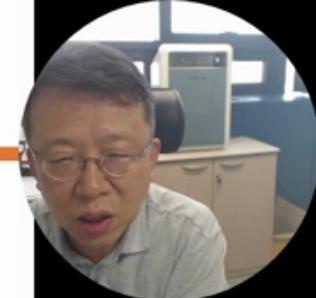
- 프로세스: 흔히 일을 처리하는 과정 또는 순서
- 주어진 일을 해결하기 위한 목적으로 그 순서가 정해져 수행되는 일련의 절차
- 어떤 일을 해결하고자 할 때는 해당 프로세스만 잘 따르면 목적을 달성할 수 있음
 - 예) 요리, 빨래, 소프트웨어 설치 등...



그림 1-9 요리 프로세스

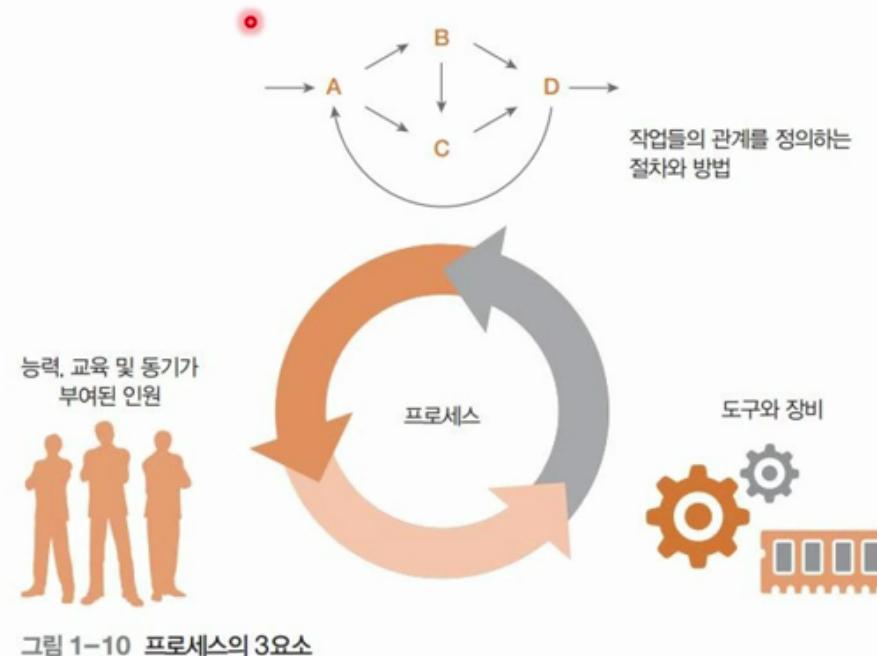


02. 소프트웨어 개발 프로세스

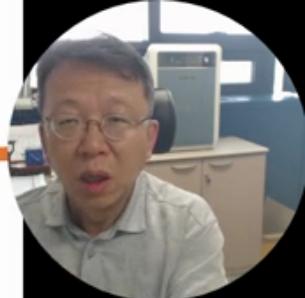


■ 소프트웨어 개발 프로세스

- 소프트웨어 개발 프로세스의 의미
 - 좁은 의미: 소프트웨어 제품을 개발할 때 필요한 절차나 과정
 - 넓은 의미: 작업을 수행하는 데 필요한 방법과 도구를 비롯해 개발과 관련된 절차를 따라 작업을 수행하는 참여자 포함



02. 소프트웨어 개발 프로세스



■ 소프트웨어 개발 프로세스 모델

- 소프트웨어 개발 프로세스 모델의 정의
 - 소프트웨어 개발 프로세스 모델은 소프트웨어를 어떻게 개발할 것인가에 대한 전체 흐름을 체계화한 개념
 - 개발 계획 수립부터 최종 폐기까지의 전 과정을 순차적으로 다룸
- 소프트웨어 개발 프로세스 모델의 목적
 - 주어진 예산과 자원으로 개발하고 관리하는 방법을 구체적으로 정의
 - 고품질의 소프트웨어 제품을 만드는 것이 목적
- 소프트웨어 개발 프로세스 모델의 역할
 - 프로젝트에 대한 전체적인 기본 골격을 세워 일정 계획을 수립할 수 있고, 개발 비용 산정뿐 아니라 여러 자원을 산정하고 분배할 수 있음
 - 참여자 간에 의사소통의 기준을 정할 수 있고 용어의 표준화를 가능하게 함
 - 개발 진행 상황도 명확히 파악할 수 있고 각 단계별로 생성되는 문서를 포함한 산출물을 활용해 검토할 수 있게 함



03. 주먹구구식 모델



■ 주먹구구식 모델

- 주먹구구식 모델이란
 - 공식적인 가이드라인이나 프로세스가 없는 개발 방식
 - 즉흥적 소프트웨어 개발 또는 코딩과 수정 모델이라고도 함
 - 코드를 작성해 제품을 만든 후에 요구분석, 설계, 유지보수에 대해 생각
 - 첫 번째 버전의 코드를 작성해 제품을 완성한 뒤 작성된 코드에 문제가 있으면 수정해서 해결하고 문제가 없으면 사용
- 주먹구구식 모델의 단점
 - 정해진 개발 순서나 각 단계별로 문서화된 산출물이 없어 관리 및 유지보수가 매우 어려움
 - 프로젝트 전체 범위를 알 수 없을 뿐더러 좋은 아키텍처를 만들 수도 없음
 - 개발자가 일을 효과적으로 나눠 할 수도 없음
 - 프로젝트 진척 상황을 거의 파악할 수 없음
 - 코딩을 먼저 하므로 계속 수정할 가능성이 높은데, 여러 번 수정하다 보면 가독성이 높았던 프로그램의 구조가 나빠져 수정이 매우 어려워짐

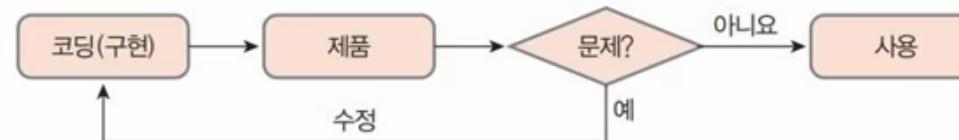


그림 1-11 주먹구구식 모델



04. 선형 순차적 모델



■ 선형 순차적 모델(폭포수 모델)

- 선형 순차적 모델 폭포에서 물이 떨어지듯이 다음 단계로 넘어가는 모델 (고전적 생명 주기)
- 소프트웨어 공학의 대명사로 여겨질 만큼 초기에 개발된 전통적인 모델
- 공장 생산 라인의 작업 프로세스와 유사한데, 표준 프로세스를 정해 소프트웨어를 순차적으로 개발

■ 폭포수 모델의 개발 절차

- 폭포의 물이 위에서 아래로 떨어지듯이 계획, 분석, 설계, 구현, 테스트, 유지보수의 각 단계가 하향식으로 진행
- 각 단계가 끝날 때마다 확실히 매듭을 짓고 그 결과를 확인한 후에 다음 단계로 나아감
- 폭포수 모델을 적용한 개발 절차에서는 요구사항 분석 단계가 끝나면 '요구분석명세서'라는 문서를 작성
- 명세서를 기준으로 사용자에게 이상 유무를 확인받고 다음 단계인 설계 절차로 넘어감



04. 선형 순차적 모델



■ 폭포수 모델의 장점과 단점

- 폭포수 모델의 장점

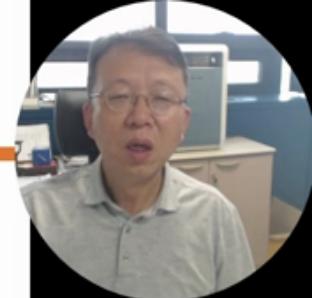
- 관리가 용이
- 체계적으로 문서화할 수 있음
- 요구사항의 변화가 적은 프로젝트에 적합함

- 폭포수 모델의 단점

- 각 단계는 앞 단계가 완료되어야 수행할 수 있음
- 각 단계마다 작성된 결과물이 완벽한 수준으로 작성되어야 다음 단계에 오류를 넘겨주지 않음
- 사용자가 중간에 가시적인 결과를 볼 수 없어 답답해 할 수 있음



04. 선형 순차적 모델



■ V 모델

- V 모델

- 폭포수 모델의 변형으로, 테스트 단계를 추가 확장해 테스트 단계가 분석 및 설계와 어떻게 관련되어 있는지를 나타냄
- 폭포수 모델이 산출물 중심이라면 V 모델은 각 개발 단계를 검증하는 데 초점을 두므로 오류를 줄일 수 있음

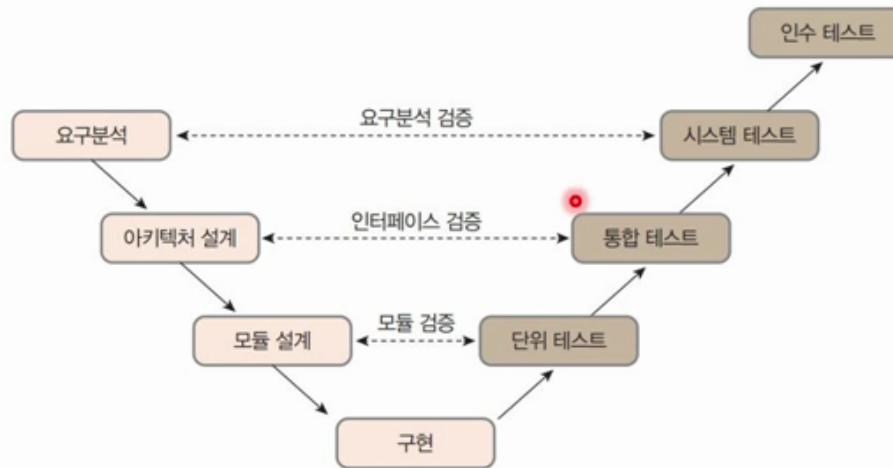


그림 1-14 V 모델

- V 모델에서 분석·설계 단계는 왼쪽에 나타내고, 테스트는 오른쪽에 나타냄
- 구현은 V 모양의 꼭짓점에 위치



05. 진화적 프로세스 모델



■ 진화적 프로세스 모델

- 진화적 프로세스 모델
 - 새로운 요구가 수시로 발생해 이에 민첩하게 대응할 수 있는 방법
 - 초기의 사용자 요구에 따라 가상으로 실행되는 초기 버전의 프로토타입을 작성
 - 사용자는 사용자 인터페이스 중심의 화면과 실행 후 나타나는 가상의 결과 화면을 확인
 - 변경된 요구사항을 반영하거나 추가해 2차 프로토타입을 만들어 사용자에게 보여줌, 이 과정을 반복

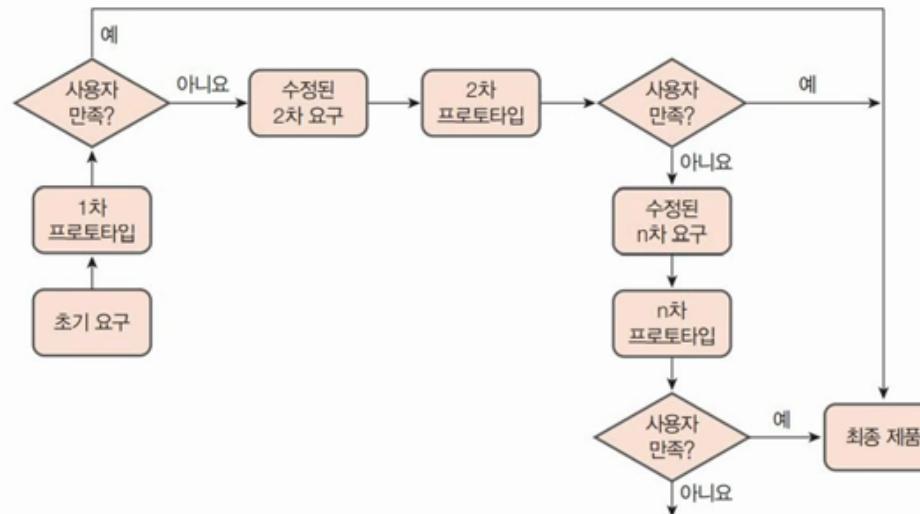


그림 1-15 진화적 프로세스 모델



05. 진화적 프로세스 모델



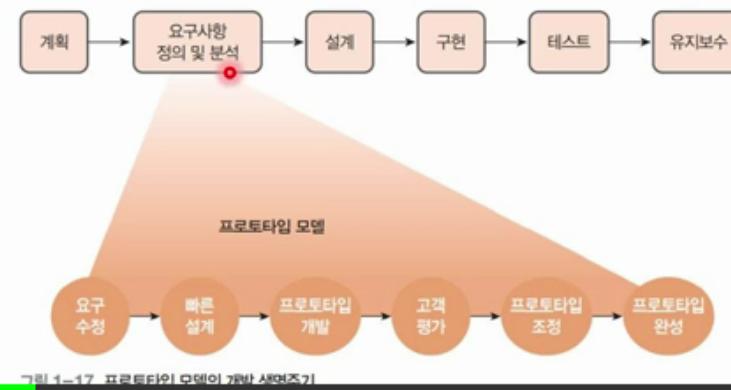
■ 프로토타입 모델

- 프로토타입 모델의 정의

- 사전적 의미는 대량 생산에 앞서 미리 제작해보는 원형 또는 시제품
- 소프트웨어 개발에서는 정식 절차에 따라 완전한 소프트웨어를 만들기 전에 사용자의 요구대로 일단 모형을 만들고 사용자와 의사 소통하는 도구로 활용
- 예) 아파트 모델하우스

- 프로토타입 모델의 개발

- 프로토타이핑: 프로토타입을 만드는 것
- 개발자는 사용자의 초기 요구사항을 반영해 1차 프로토타입(입출력 화면)을 만들고 이를 사용자에게 보여줌
- 사용자는 1차 프로토타입을 보면서 추가 요구나 수정 요구를 하고, 개발자는 이를 받아들여 2차 프로토타입을 제작
- 프로토타입 모델은 사용자의 요구가 불투명하고 요구사항의 변화가 계속 많이 발생하는 경우에 적합



05. 진화적 프로세스 모델



■ 프로토타입 모델

▪ 프로토타입 모델의 장점과 단점

• 프로토타입 모델의 장점

- 프로토타입이 개발자와 사용자 간의 의사소통 도구로 사용되어 구체적이고 원활하게 대화할 수 있음
- 요구사항을 여러 번 반복하는 과정을 통해 사용자의 요구가 충분히 반영된 요구분석명세서를 만들 수 있음
- 개발되는 소프트웨어가 어떤 모습인지 예측할 수 있으므로 개발 초기에 만족감을 가져 개발에 적극적으로 참여하려는 의지를 보임
- 사용자의 요구가 충분히 반영되어 최종 제품이 나오므로 유지보수에 필요한 노력과 시간을 많이 줄일 수 있음

• 프로토타입 모델의 단점

- 반복적인 소프트웨어 개발 단계로 인해 필요한 투입 인력과 비용 산정이 어려움
- 개발된 프로토타입으로는 완전히 동작할 수 없는데도 빠른 시간 안에 최종 결과가 나올 것이라고 사용자가 착각할 수 있음
- 중간 점검을 할 수 있는 이정표나 산출물을 생성할 수 없어 프로토타이핑 과정을 관리·통제하기 어려움
- 개발 범위가 명확하지 않아 소프트웨어 개발의 목표나 종료 시점이 불명확해질 수 있음
- 프로토타입에 따른 추가 비용이 들 수 있음



05. 진화적 프로세스 모델



o

■ 프로토타입 모델

▪ 프로토타입 모델의 개발 절차

① 요구사항 정의 및 분석

- 폭포수 모델과 달리 프로토타입 모델에서는 1차로 개략적인 요구사항을 정의한 후 2차, 3차, ... n차를 반복하면서 완성도를 높여 최종 프로토타입을 개발

② 프로토타입 설계

- 동작이 가능하도록 최종 코딩을 할 수 있는 설계가 아니라, 입력 화면과 출력 화면을 통해 사용자가 만들어질 시스템이 어떻게 수행되는지 파악할 수 있도록 사용자 인터페이스를 중심으로 설계

③ 프로토타입 개발

- 프로토타입 개발은 프로토타입 모델의 취지에 맞게 가상 수행을 전제로 한 실행을 보여주는 것이 우선

④ 사용자에 의한 프로토타입 평가

- 프로토타입 모델에서 매우 중요한 단계
- 사용자는 1차로 개발된 프로토타입을 보며 요구사항이 충실히 반영되었지를 확인
- 사용자는 확 인이 끝나면 추가 및 수정 요구사항을 전달
- 개발자는 이를 반영해 2차 설계를 한 뒤 그에 따른 2차 프로토타입을 개발
- 같은 과정을 n번 반복해 사용자의 추가 요구사항이 더 이상 없을 때 최종 프로토타입이 만들어짐



05. 진화적 프로세스 모델



■ 나선형 모델

▪ 나선형 모델

- 원래 보엠이 제안
- 개발 과정이 뱅글뱅글 돌아 점점 완성도가 높은 제품이 만들어짐
- 나선형 모델의 위험 분석 단계에서 위험 요소는 소프트웨어 개발 과정이 순조롭게 진행되는 데 방해되는 모든 것
- 초기 요구분석 후 프로토타입 개발 이전에 위험 분석 단계를 거침

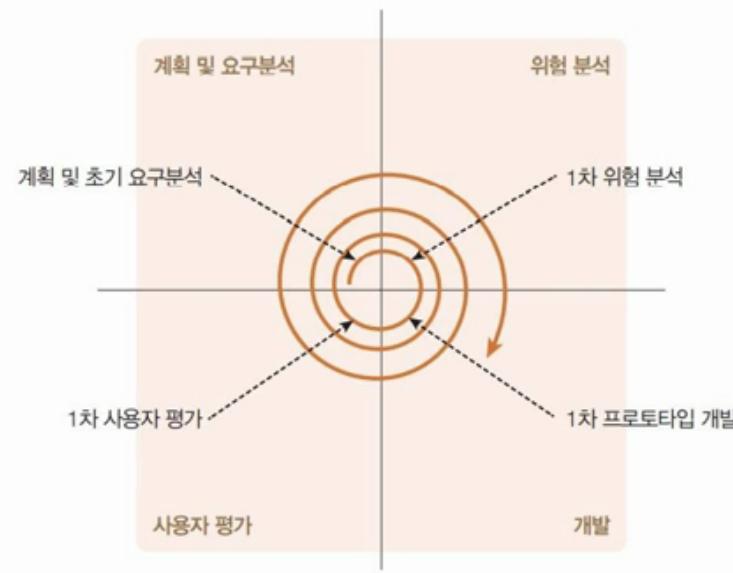


그림 1-18 나선형 모델



05. 진화적 프로세스 모델



■ 나선형 모델

▪ 나선형 모델의 개발 절차

① 계획 및 요구분석 단계

- 사용자의 개발 의도를 파악해 해당 프로젝트의 목표를 명확히 하고, 여러 제약 조건의 대안을 고려한 계획을 수립
- 사용자의 요구를 통해 파악 한 기능 요구사항과 성능 같은 비기능 요구사항을 정의하고 분석

② 위험 분석 단계

- 프로젝트 수행에 방해되는 위험 요소를 찾아 목록을 작성하고 위험에 대한 예방 대책을 논의
- 심각한 위험이 존재하는 경우에는 해당 프로젝트를 계속 진행해도 되는지를 결정
- 위험요소 – 프로젝트 수행 중 개발자 이직, 요구사항 확정 이후 계속되는 변경 요구, 프로젝트 수행 중 발주사의 경제적 어려움, 처음 예측 인력보다 추가 인력 필요, 처음 예측 개발 기간 초과, 처음 예측 개발비로 완료할 수 없는 경우 등

③ 개발 단계

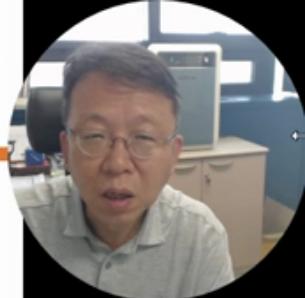
- 프로토타입을 만드는데, 다른 소프트웨어 개발 프로세스의 설계와 구현에 해당

④ 사용자 평가 단계

- 사용자가 만족할 때까지 n번 반복해 더 이상의 추가 및 수정 요구가 없으면 최종 제품을 개발
- 사용자 평가 단계는 진화적 프로토타입 모델에서 매우 핵심적이고 중요



05. 진화적 프로세스 모델



■ 나선형 모델

▪ 나선형 모델의 장점과 단점

• 나선형 모델의 장점

- 전에 위험을 의식하고 개발하기 때문에 프로젝트가 중단되는 심각한 사태가 일어날 확률이 비교적 적음
- 사용자 평가가 반영된 반복적 개발 방식에 의해 사용자 요구가 충분히 반영되어 사용자의 불만이 적음

• 나선형 모델의 단점

- 요구분석, 위험 분석, 개발, 사용자 평가가 반복적으로 계속 진행되기 때문에 프로젝트 기간이 길어질 수 있음
- 반복 횟수가 많아질수록 프로젝트 관리가 어려움
- 위험 관리가 중요한 만큼 위험 관리 전문가가 필요하다는 부담감



06. 단계적 개발 모델



■ 단계적 개발 모델

▪ 단계적 개발 모델

- 개발자가 먼저 릴리스 1을 개발해 사용자에게 제공하면 사용자가 이를 사용
- 사용자가 릴리스 1을 사용하는 동안 개발자는 다음 버전인 릴리스 2를 개발
- 개발과 사용을 병행하는 과정을 반복해 진행하면서 완료
- 릴리스를 구성하는 방법에 따라 점증적 개발 방법과 반복적 개발 방법으로 나뉨

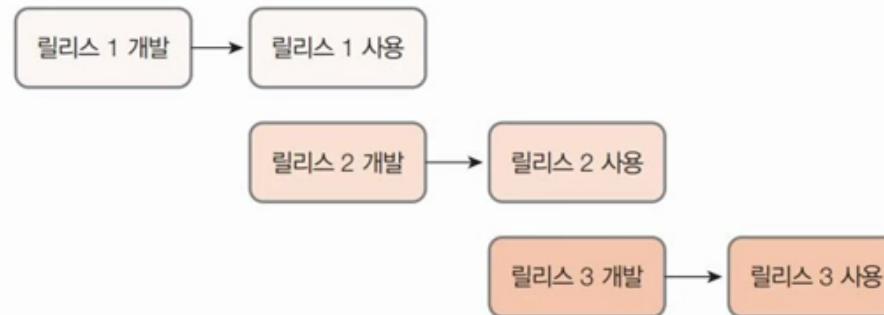


그림 1-19 단계적 개발 모델



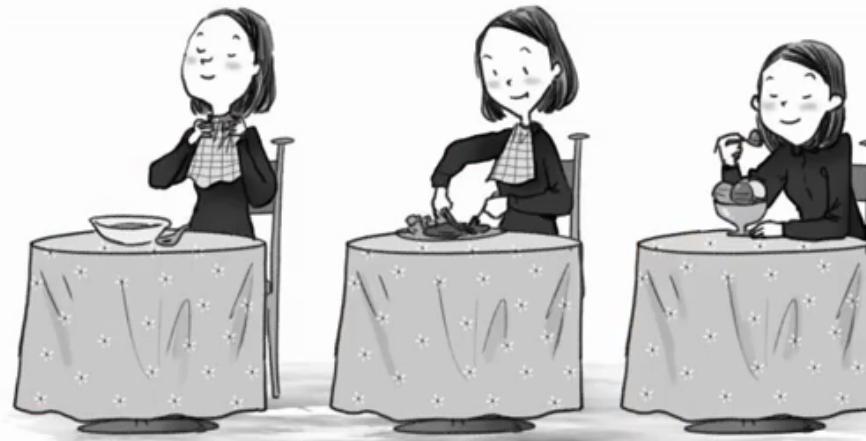
06. 단계적 개발 모델



■ 단계적 개발 모델

▪ 점증적 개발 방법: 개발 범위의 증가

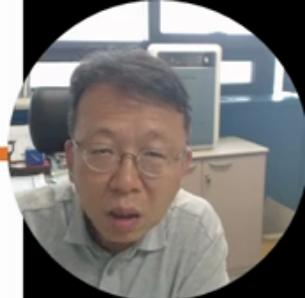
- 중요하다고 생각되는 부분부터 차례로 개발한 후 그 일부를 사용하면서 개발 범위를 점차 늘려 가는 방식
- 예) 양식 코스 요리는 전채 요리-메인 요리-후식으로 하나가 끝나면 또 하나를 먹음



● 그림 1-20 점증적 개발 방법의 예: 양식 코스 요리



06. 단계적 개발 모델



■ 단계적 개발 모델

▪ 점증적 개발 방법: 개발 범위의 증가

- 점증적 방법 예 1: 도서 집필 시

- 총 10장으로 구성된 소프트웨어 공학 책을 집필할 때 1장을 완벽히 쓰고, 2장, 3장, ..., 10 장까지 완성해 나가는 방식
 - 1장부터 순차적으로 집필하지 않고 저자가 가장 중요하다고 생각하는 부분(또는 원고가 준비된 순서)부터 원고를 집필

- 점증적 방법 예 2: 3층 건물 건축 시

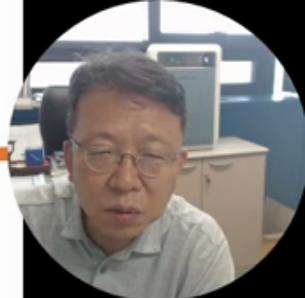
- 3층짜리 건물을 지을 때 1층을 먼저 완벽하게 지은 후 1층만 실제로 사용
 - 그러다 경제적으로 여건이 갖추어 졌을 때 2층을 올린다. 같은 방법으로 마지막 3층을 짓고 목표한 건물을 완성
 - 단순히 1층짜리 건물을 지을 때 보다는 비용이 많이 듈다는 단점

- 점증적 방법 예 3: 대학 종합정보시스템 개발 시

- 대학에서 사용하는 종합정보시스템을 개발할 경우, 가장 중요한 교무/학사 관련 시스템을 먼저 개발해 사용
 - 회계 업무를 비롯한 다른 부서의 업무 시스템을 차츰 개발 해나가는 방식



06. 단계적 개발 모델



■ 단계적 개발 모델

▪ 점증적 개발 방법: 개발 범위의 증가

• 점증적 방법 – 소프트웨어 개발

- 요구분석명세서에 명시된 시스템 전체를 기능에 따라 독립성 높은 서브 시스템으로 분할
- 각 서브시스템을 단계적으로 하나씩 릴리스해 완성하는 방법

• 점증적 방법의 장점과 단점

- 한꺼번에 많은 비용을 들이지 않아도 됨
- 완전히 새로운 시스템 전체를 한 번에 주었을 때 조직이 받는 충격을 완화할 수 있음
- 소프트웨어를 단계적으로 도입하면 조직에 자연스럽게 변화를 줄 수 있음
- 이미 사용하고 있는 서브시스템이 있어 어떤 유형으로 개발해야 하는지 잘 알 수 있음
- 서브시스템들이 서로 관련이 있어 처음 설계할 때부터 이후에 개발할 다른 서브시스템과의 연관성을 고려해야 함
- 이미 개발된 서브시스템들과 통합하는 데 어려움을 겪을 수 있음



06. 단계적 개발 모델



■ 단계적 개발 모델

▪ 반복적 개발 방법: 품질의 증가

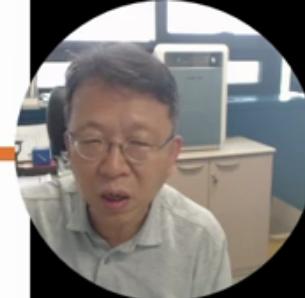
- 초기에 시스템 전체를 일차적으로 개발해 인도한 후, 각 서브시스템의 기능과 성능을 변경 및 보강해 완성도를 높임
- 초기의 요구사항이 불분명한 경우에 적합
- 예) 한정식을 먹을 때는 모든 음식을 한 상에 가득 차려놓고 조금씩 모두 맛본 후 맛있는 음식은 여러 번 먹음



그림 1-21 반복적 개발 방법의 예: 한 상 가득 차려진 한정식



07. 통합 프로세스 모델



■ 통합 프로세스 모델

▪ 반복적 개발 방법론의 등장

- 폭포수 모델은 각 단계별로 깔끔하게 정리 되어 있지만 사용자의 요구사항이 많으면 민첩하게 대처할 수 없음
- 폭포수 모델의 문제점을 해결하기 위해 작업을 계속해서 반복 수행하는 반복적 개발 방법론이 등장
- 통합 프로세스 모델은 반복적 생명주기를 기반으로 하는 프로세스 모델 중 가장 많이 사용

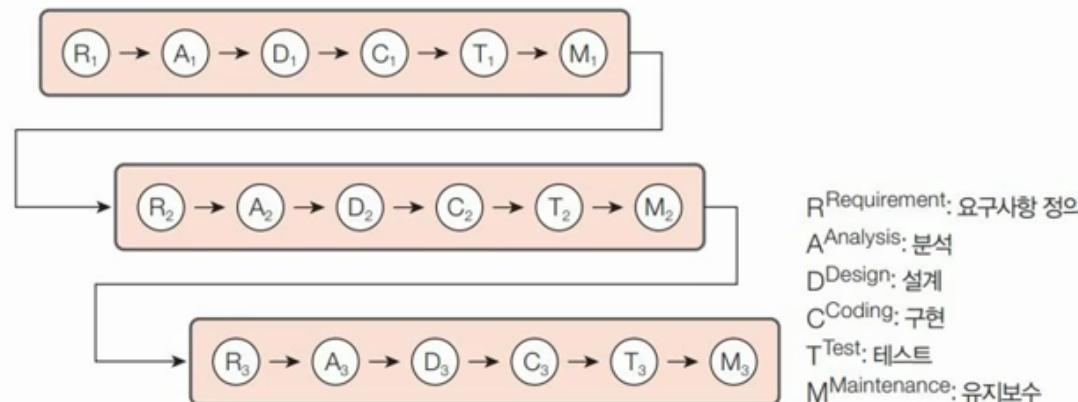


그림 1-22 반복적 개발 방법



07. 통합 프로세스 모델



■ 통합 프로세스 모델

▪ 통합 프로세스 모델

- 통합 프로세스 모델은 반복적 생명주기를 기반으로 하는 프로세스 모델 중 가장 많이 사용
- OMG 가 공개한 UML과 함께 제안되어 통합된 프로세스
- Rational사에 의해 RUP라는 이름으로 상품화

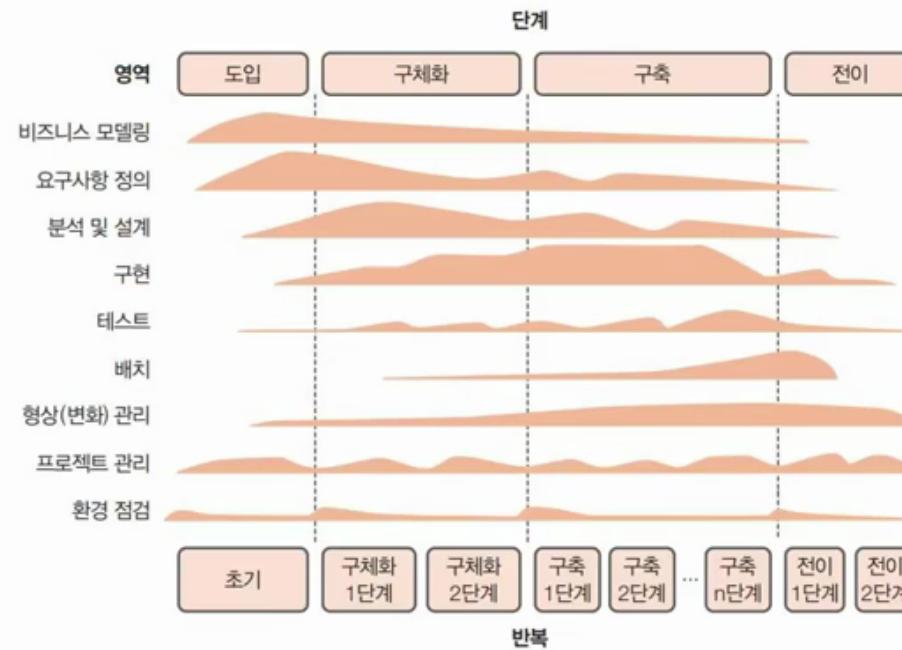


그림 1-23 통합 프로세스 모델



08. 애자일 프로세스 모델



■ 애자일 프로세스 모델의 이해

- 애자일 프로세스 모델

- 애자일(agile): '날렵한', '민첩한'
- 애자일 프로세스 모델: 고객의 요구에 민첩하게 대응하고 그때그때 주어지는 문제를 풀어나가는 방법론
- 폭포수 모델처럼 계획을 기반으로 하는 프로세스 중심의 전통적인 모델은 산출물 위주의 거대하고 무거운 방법론에 해당해 요구사항의 변화에 유연하게 대처하기 어려워 가볍고 비교적 변화를 수용하기 위한 방법론이 필요.
예로 익스트림 프로그래밍, 스크럼, 크리스털 방법론
- 애자일은 가벼운 프로세스 방법론의 공통적인 특성을 정의하는 말
- 2001년 1월에 '애자일 연합' 그룹에서 서로의 공통점을 찾아 '애자일 선언문'이라는 공동 선언서를 발표

- 애자일의 기본 가치

- 프로세스와 도구 중심이 아닌, 개개인과의 상호 소통을 중시
- 문서 중심이 아닌, 실행 가능한 소프트웨어를 중시
- 계약과 협상 중심이 아닌, 고객과의 협력을 중시
- 계획 중심이 아닌, 변화에 대한 민첩한 대응을 중시
- 환경과 고객의 변화에 능동적으로 대처하는 것을 강조

• 폭포수 모델이 계획, 프로세스, 프로젝트 관리, 문서(산출물)에 중점을 둔다면, 애자일은 고객과의 협업, 빠른 시간 안에 고객이 작동해볼 수 있는 소프트웨어, 환경과 고객의 변화에 능동적으로 대처하는 것을 강조



08. 애자일 프로세스 모델



■ 애자일 프로세스 모델의 이해

- 애자일의 원칙

- 최우선적인 목표는 고객을 만족시키기 위해 가치 있는 소프트웨어를 빨리, 지속적으로 제공하는 것
- 개발 후반에 새로 추가되는 요구사항도 기꺼이 받아들임
- 애자일 프로세스는 고객의 경쟁력을 위해 요구사항의 변경을 받아들임
- 동작 가능한 소프트웨어를 짧으면 2주, 길면 2개월 간격으로 자주 고객에게 전달, 이때 간격은 짧을수록 좋음
- 프로젝트가 진행되는 동안에 업무 담당자와 개발자는 매일 서로 의견을 주고받으며 함께 참여
- 가장 효과적인 의사소통 방법은 역시 직접 만나 얼굴을 보면서 대화하는 것
- 진척 사항의 척도를 나타내는 방법 중 가장 좋은 방법은 실행 가능한 소프트웨어를 보여주는 것
- 자율적 사고와 자유로운 분위기의 프로젝트 팀에서 최고의 아키텍처, 요구사항, 설계가 나옴
- 프로젝트의 효율성을 향상하기 위해 개발 팀 스스로 정기적인 미팅을 진행해 자신들의 행동을 되돌아보고 조율 및 수정

- 애자일 프로세스 개발 방법

- 가장 기본이 되는 기능만 1차 요구사항으로 분석하고 이를 반복으로 나누어 개발
- 사용자가 릴리스 1을 사용하는 동안 개발자는 2차 개발을 진행
- 이때 좀 더 복잡한 편집 기능과 고급 기능을 추가해 마찬가지로 이를 반복으로 나누어 개발
- 프로세스 개발 방법은 반복적인 개발을 통한 잣은 출시를 목표로 함
- 실행 가능한 프로토타입을 만들어 사용자에게 확인 받음
- 좀 더 빠른 시간 안에 일부지만 소프트웨어를 사용할 수 있게 하는 것을 중요하게 생각



08. 애자일 프로세스 모델

표 1-1 애자일 프로세스 모델과 폭포수 모델의 비교

구분	애자일 프로세스 모델	폭포수 모델
추가 요구사항의 수용	처음 수집한 요구사항을 전체 중 일부로 인정하고 시작하므로 언제든지 추가 요구사항이 있을 것으로 간주한다. 따라서 추가 요구사항을 수용할 수 있는 방법으로 설계되어 있다.	요구사항 분석이 완전히 완료된 후에 설계 단계로 넘어가므로 새로운 요구사항을 추가하기 쉽지 않다. 추가 요구사항을 반영하기 어려운 구조다.
릴리스 시점	가능하면 자주, 빨리 제품에 대한 프로토타입을 만들어 사용자에게 보여준다. 이러한 방식을 반복적으로 수행해 최종 제품을 만들기 때문에 자주 릴리스 된다.	요구사항에 대한 분석, 설계, 구현 과정이 끝나고 최종 완성된 제품을 릴리스한다.
시작 상태	계속적인 추가 요구사항을 전제로 하는 방식이라 시작 단계에서는 부족한 점이 많지만 점차 완성도가 높아진다.	한 번 결정된 단계는 그 이후에 변동이 적어야 한다. 따라서 완성도를 최대한 높여 다음 단계로 넘어가기 위해 시작 단계의 완성도가 매우 높다.
고객과의 의사소통	사용자와 함께 일한다는 개념을 담고 있다. 처음부터 사용자의 참여를 유도하고 많은 대화를 하면서 개발을 진행한다.	사용자 요구사항을 정의한 후 사용자에게 더는 추가 요구가 없다는 확답을 받고 개발에 들어간다. 산출물을 근거로 하기 때문에 사용자와의 대화는 적다.
진행 상황 점검	개발자와 사용자는 개발 초기부터 지속적으로 진행 상황을 공유하며 함께 관심을 갖고 진행해 나간다.	단계별 산출물을 중요시하기 때문에 단계별 산출물에 대한 결과로 개발의 진척 상황을 점검한다.
분석, 설계, 구현 진행 과정	분석, 설계, 구현이 하나의 단계와 그 단계 안의 반복마다 한꺼번에 진행된다. 다만 어떤 단계에서는 분석이 많고 구현이 적고 어떤 단계에서는 분석이 적고 구현이 많다는 차이만 있을 뿐이다.	분석, 설계, 구현 과정이 명확하다. 각 과정에서 생산되는 산출물 중심의 개발 방식이기 때문에 단계가 명확히 구별되어 있다. 따라서 분석이 끝난 후 설계를, 설계가 끝난 후 구현 작업을 진행한다.
모듈(컴포넌트) 통합	개발 초기부터 빈번한 통합을 통해 문제점을 빨리 발견하고 수정하는 방식을 택한다. 문제점을 빨리 발견 하므로 비용을 절감할 수 있다는 장점이 있다.	V 모델에서 설명한 것처럼 구현이 완료된 후에 모듈 간의 통합 작업을 수행한다.



08. 애자일 프로세스 모델



■ 애자일 프로세스 모델: 스크럼

- 스크럼 방식

- 스크럼

- 원래 럭비 경기에서 사용하는 용어로 반칙으로 인해 경기가 중단되었을 때 쓰는 대형
 - 팀이라는 단어가 주는 의미를 개발 팀에 적용해 효율적인 성과를 얻기 위해 소프트웨어 개발 프로세스에서 사용

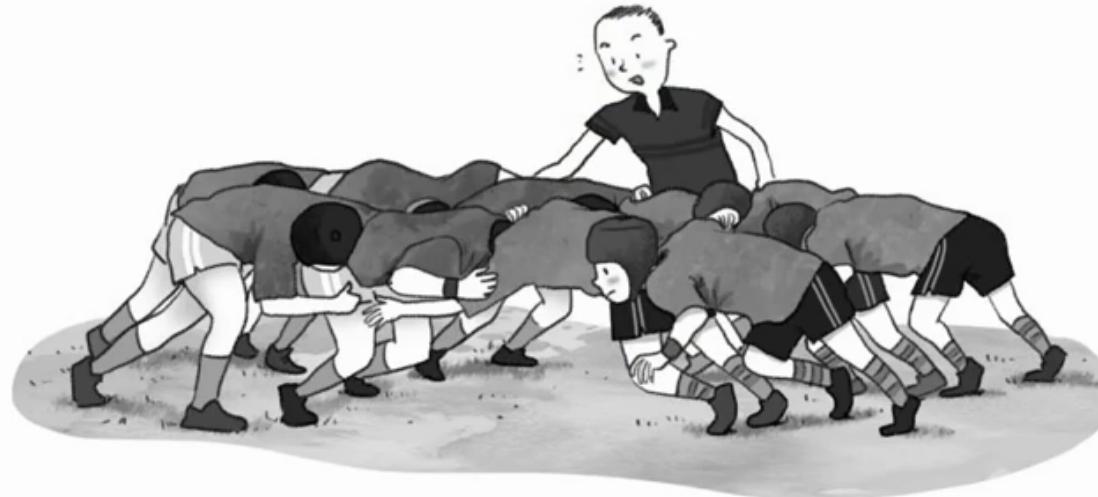


그림 1-29 럭비 경기의 스크럼 대형



08. 애자일 프로세스 모델



■ 애자일 프로세스 모델: 스크럼

▪ 스크럼 방식

- 스크럼 방식
 - 소프트웨어 개발보다는 팀의 개선과 프로젝트 관리에 중점을 둔 애자일 방법론
 - 경험적 관리 기법 중 하나
 - 구체적인 프로세스를 명확하게 제시하지 않음



그림 1-30 스크럼 방식



08. 애자일 프로세스 모델



■ 애자일 프로세스 모델: 스크럼

▪ 스크럼 방식

표 1-2 스크럼 방식의 진행 과정

단계	수행 목록	내용
1	제품 기능 목록 작성	<ul style="list-style-type: none">요구사항 목록에 우선순위를 매겨 제품 기능 목록 작성
2	스프린트 계획 회의	<ul style="list-style-type: none">스프린트 구현 목록 작성스프린트 개발 시간 추정
3	스프린트 수행	<ul style="list-style-type: none">스프린트 개발일일 스크럼 회의스프린트 현황판 변경소멸 차트 표시
4	스프린트 개발 완료	<ul style="list-style-type: none">실행 가능한 최종 제품 생산
5	스프린트 완료 후	<ul style="list-style-type: none">스프린트 검토 회의스프린트 회고두 번째 스프린트 계획 회의



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 제품 기능 목록

• 제품 기능 목록의 정의

- 사용자가 요구하는 제품의 기능 목록을 말하며, 제품에 관한 모든 요구사항에 대해 우선순위가 정해져 있음
- 우선순위는 고객 측 대표인 제품 책임자가 결정하며 사용자와 계속 미팅하면서 목록이 완성
- 한 번 결정된 제품 기능 목록은 확정된 것이 아니고 개발 중이라도 수정이 가능
- 일반적으로 한 주기가 끝날 때까지는 제품 기능 목록을 수정하지 않음

표 1-3 제품 기능 목록의 예: 학사관리시스템

순위	기능	사용자 스토리	포인트	중요도
4	교과목등록	직원은 교과목등록 기간을 설정한다.	4	중
		교수는 교과목을 등록, 수정, 삭제한다.	7	
		사용자(교수/학생/직원/조교)는 등록된 교과목을 조회한다.	5	
3	개설과목등록	직원은 개설과목 기간을 설정한다.	4	중
		교수는 개설과목을 등록, 수정, 삭제한다.	7	
		사용자는 등록된 개설과목을 ● 한다.	5	
1	수강신청	직원은 수강신청 기간을 설정한다.	4	상
		직원은 수강신청 변경 기간을 설정한다.	4	
		학생은 수강신청 과목을 등록, 수정, 삭제한다.	7	
		사용자는 학생의 수강신청 내역을 조회한다.	5	
2	성적등록	직원은 성적등록 기간을 설정한다.	4	중
		교수는 성적을 입력, 수정, 삭제한다.	7	
		사용자는 성적을 조회한다.	5	
...



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 제품 기능 목록

• 사용자 스토리

- 제품 기능이 도출되면 각 기능을 간략하게 서술
- 주로 한 장의 인덱스 카드(포스트잇)로 작성
- 기능을 상세하게 적는 것이 아니라 개발자와 대화를 지속할 수 있는 단서 정도로만 활용할 수 있게 작성
- 론 제프리스가 제시한 사용자 스토리의 구성 요소
 - 카드: 기능을 서술하는 것으로 포스트잇을 많이 사용
 - 대화: 사용자와 개발자가 충분한 대화를 통해 요구사항을 도출
 - 확인: 스토리가 완료되는 조건을 서술
- (マイ크 콘: mike Cohn)의 사용자 스토리의 작성 기준
 - 스토리끼리는 서로 의존적이지 않아야 함
 - 사용자 스토리는 언제든 변경할 수 있어야 하므로 자세히 서술할 필요가 없음
 - 사용자 스토리는 사용자의 이야기이므로 사용자가 이해할 수 있는 언어와 용어로 작성해야 함
 - 또한 사용자에게 필요한 이야기이므로 개발자에게 필요한 내용을 서술해서는 안 됨
 - 사용자 스토리를 보면 개발 규모가 어느 정도이고 투입 공수는 얼마나 필요할지 개발자들이 짐작할 수 있어야 함
 - 개발 규모가 적절해야 함
 - 사용자 스토리의 구성 요소 확인을 위해 테스트가 가능한 테스트 케이스를 만들어야 함. 예를 들어, '수강신청 시 속도가 빨라야 한다'는 정확한 기준이 없어 테스트할 수 없으니, '수강신청 시 동시 접속자 수 2,000명까지는 1기가바이트 속도를 유지해야 한다'처럼 구체적으로 작성.



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 제품 기능 목록

• 제품 기능 목록의 정의

- 사용자가 요구하는 제품의 기능 목록을 말하며, 제품에 관한 모든 요구사항에 대해 우선순위가 정해져 있음
- 우선순위는 고객 측 대표인 제품 책임자가 결정하며 사용자와 계속 미팅하면서 목록이 완성
- 한 번 결정된 제품 기능 목록은 확정된 것이 아니고 개발 중이라도 수정이 가능
- 일반적으로 한 주기가 끝날 때까지는 제품 기능 목록을 수정하지 않음

표 1-3 제품 기능 목록의 예: 학사관리시스템

순위	기능	사용자 스토리	포인트	중요도
4	교과목등록	직원은 교과목등록 기간을 설정한다.	4	중
		교수는 교과목을 등록, 수정, 삭제한다.	7	
		사용자(교수/학생/직원/조교)는 등록된 교과목을 조회한다.	5	
3	개설과목등록	직원은 개설과목 기간을 설정한다.	4	중
		교수는 개설과목을 등록, 수정, 삭제한다.	7	
		사용자는 등록된 개설과목을 조회한다.	5	
1	수강신청	직원은 수강신청 기간을 설정한다.	4	상
		직원은 수강신청 변경 기간을 설정한다.	4	
		학생은 수강신청 과목을 등록, 수정, 삭제한다.	7	
		사용자는 학생의 수강신청 내역을 조회한다.	5	
2	성적등록	직원은 성적등록 기간을 설정한다.	4	중
		교수는 성적을 입력, 수정, 삭제한다.	7	
		사용자는 성적을 조회한다.	5	
...



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 제품 기능 목록

- 사용자 스토리

- 작성된 사용자 스토리는 스토리 보드에 나열되고 우선순위와 스토리 포인트가 결정

- 스토리 포인트 산정

- 스토리 포인트: 요구사항의 규모를 측정하는 단위로 업무량을 이용해 산정
 - 스토리 간의 상대적인 업무량을 비교해 가장 적을 때를 1로 두고 이를 기준으로 얼마나 큰지에 따라 스토리 포인트를 산정
 - 일반적인 소프트웨어 개발 방법론에서는 개발에 소요되는 시간을 일/주/월의 시간 단위로 예측
 - 애자일 방법론에서는 스토리 포인트라는 추상 개념으로 예측



그림 1-31 스토리 보드



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 계획 수립

- 스프린트: 작업량이 그렇게 많지 않고 개발 기간도 짧은 경우를 의미
- 작은 단위의 개발 업무를 단기간 내에 전력 질주해 수행
- 하나의 스프린트에 어떤 사용자 스토리를 몇 개 포함할 것인지는 스프린트 계획 회의에서 결정
- 보통 1~4주 정도를 하나의 스프린트로 봄
- 외부의 개발 방해 요소를 차단하는 것은 스크럼 마스터의 역할
- 하나의 스프린트 개발이 끝나면 사용자에게 시연하고 사용자는 피드백을 제공
- 계획된 일정 안에 개발을 마치지 못해도 정해진 일정이 끝나면 하나의 스프린트가 끝남



그림 1-32 스프린트



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 계획 수립

• 스프린트 계획 회의

- 전체적인 스프린트 계획 회의
 - 사용자의 대변인격인 제품 책임자를 통해 사용자가 원하는 것이 무엇인지를 파악하는 데 중점을 둠
 - 이를 위해 스크럼 마스터는 제품 기능 목록을 검토하면서 어떤 항목을 가장 높은 순위로 놓았는지 확인
 - 그 배경과 목표에 대해 팀원들과 토의하며 제품 책임자의 의도를 파악
- 세부적인 스프린트 계획 회의
 - 우선순위가 높은 항목을 어떻게 구현할 것인지 구체적인 작업 계획을 설립
 - 먼저 우선순위가 매겨진 사용자 요구사항 목록인 제품 기능 목록에서 개발 항목을 결정
 - 스프린트 구현 목록을 작성한다. 여기에 팀원들은 정해진 작업을 수행하는 데 소요되는 시간을 추정



08. 애자일 프로세스 모델

■ 스크럼 방식의 이해

■ 스프린트 계획 수립

• 스프린트 구현 목록 작성

- 제품 기능 목록에 있는 스토리 중에서 선택해 작성
- 스프린트 구현 목록은 스프린트 계획 회의에서 결정
- 기준은 기간 내에 완료할 수 있는 만큼의 사용자 스토리
- 실행 가능한 결과가 나올 수 있는 수준에서 결정

표 1-4 스프린트 구현 목록의 예

스프린트	사용자 스토리	SP	작업	MD	개발자
수강신청	학생은 수강신청 과목을 등록/수정/삭제/조회할 수 있다.	20	UI 설계 웹페이지 설계 장바구니 구현 DB 테이블 설계 단위 테스트 코드 검토	1 2 2 3 1 1	김희석
성적등록	교수는 성적을 등록/수정/삭제/조회할 수 있다.	16	UI 설계 웹페이지 설계 DB 테이블 설계 단위 테스트 코드 검토	1 2 2 1 1	서웅식
교과목, 개설 과목 등록	교수는 교과목을 등록/수정/삭제/조회할 수 있다.	9	UI 설계 웹페이지 설계 DB 테이블 설계 단위 테스트 코드 검토	1 2 2 3 1	박양구
	교수는 개설과목을 등록/수정/삭제/조회할 수 있다.	9	UI 설계 웹페이지 설계 DB 테이블 설계 단위 테스트 코드 검토	1 2 2 3 1	이진현
...

※ SP: 스토리 포인트, MD: 연 작업 시간



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 수행

• 소멸 차트

- 계획 대비 작업이 어떻게 진행되고 있는지를 날짜별 남은 작업으로 나타냄
- 소멸 차트는 개발 후 남은 작업량을 표현하므로 시간이 지남에 따라 차트가 감소

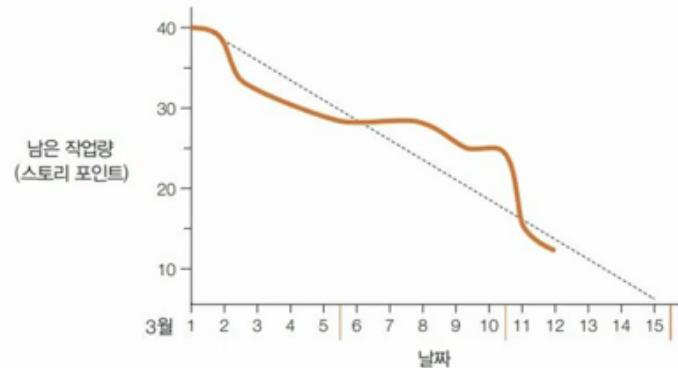


그림 1-33 소멸 차트

- 가로축: 시간 축으로 스프린트 반복 주기 날짜 수
- 세로축: 완료된 작업의 추정 일수(스토리 포인트로 표현)
- 계획 그래프: 처음 계획을 세웠을 때 날짜별로 남은 작업량(점선으로 표시)
- 실제 그래프: 작업을 수행하면서 날짜별로 실제 남은 작업량(실선으로 표시)



08. 애자일 프로세스 모델

■ 스크럼 방식의 이해

▪ 스프린트 수행

- 일일 스크럼 회의

- 스프린트 기간에 하는 회의로 다음과 같은 특징이 있음
 - 매일 함
 - 서서 함
 - 약속된 시간에 함
 - 짧게(15분 정도) 함
 - 진행 상황만 점검
 - 스프린트 작업 목록을 잘 개발하고 있는지 확인
 - 모든 팀원이 참석
 - 한 사람씩 어제 한 일을 얘기
 - 한 사람씩 오늘 할 일을 얘기
 - 한 사람씩 문제점 및 어려운 점 정도만 얘기
 - 매일 완료된 세부 작업 항목을 완료 상태로 옮겨 스프린트 현황판을 업데이트
 - 개별 팀원에 대한 진척 상태를 확인
 - 그날의 남은 작업량을 소멸 차트에 표시

- 스크럼 마스터의 역할

- 팀원들의 개발 작업에 방해되는 요소를 찾아 문제를 해결
- 개발자 개개인이 수행하고 있는 일의 진행 상황을 확인
- 소멸 차트에 그날의 남은 작업량을 표시



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 수행

• 스프린트 현황판

- 개발 팀의 개발 현황(진척도, 남은 작업, 진행 속도)을 나타냄

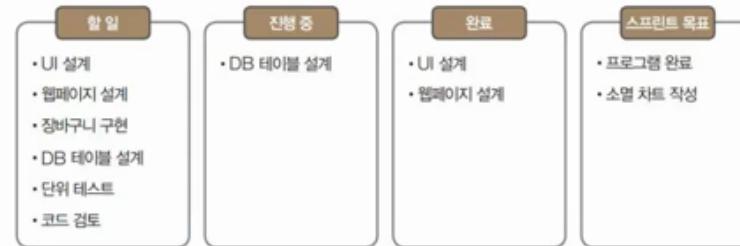


그림 1-34 스프린트 현황판의 예

• 스프린트 진척 관리

- 스프린트의 각 작업에 대한 날짜별 작업 시간과 남은 시간 등을 기록해 진척 관리를 할 수 있음

표 1-5 수강신청 스프린트의 진척 관리표

스프린트	작업	1일	2일	3일	4일	5일	6일	7일	8일	9일	10일
수강신청	UI 설계	1									
	웹페이지 설계		1	1							
	장바구니 구현				1	1					
	DB 테이블 설계						1	1	1		
	단위 테스트								1		
	코드 검토							●			1
총 남은 시간		9	8	7	6	5	4	3	2	1	0



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 개발 완료

- 스크럼 진행 방식을 정리하면 처음에 제품 책임자를 중심으로 제품 기능 목록을 작성
 - 스프린트 계획 회의에서 스프린트 구현 목록을 작성하고 일일 스크럼 회의를 통해 스프린트를 개발
 - 그 결과 스프린트 개발이 완료되고 최종 제품이 생산
-
- 최종 제품
- 모든 스프린트 주기가 끝나면 제품 기능 목록에서 개발하려고 했던 최종 제품이 완성



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스프린트 개발 완료 후

• 스프린트 검토 회의

- 하나의 스프린트 반복 주기(2~4주)가 끝났을 때 생성되는 실행 가능한 제품을 검토
- 스프린트 목표를 달성했는지 작업 진행과 결과물을 확인하고, 전체 흐름을 확인해 비즈니스 가치를 점검하는 데 중점
- 스크럼 팀은 스프린트 동안 작업한 결과를 참석자(고객 포함)들에게 시연하고 요구사항에 얼마나 부합하는지 검토
- 스크럼 마스터는 스프린트 동안 잘된 점, 아쉬운 점, 개선할 점 등을 찾기 위한 회고를 진행할 수 있음
- 검토는 가능한 한 4시간 안에 마침

• 스프린트 회고

- 그동안 스프린트에서 수행한 활동과 개발한 것을 되돌아보고, 개선할 점은 없는지, 팀이 정한 규칙이나 표준을 잘 준수했는지 등을 검토
- 이때 팀의 단점을 찾기보다는 강점을 찾아 더 극대화하는 데 주안점을 둠
- 또한 문제점에 대한 해결 방안을 찾는 회의가 아니므로 문제 점을 확인하고 기록하는 정도로만 진행
- 추정 속도와 실제 속도를 비교해보고, 차이가 크면 그 이유를 분석(프로세스 품질은 측정하지 않음)



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 배포 목록 작성

- 배포 사용자에게 시스템 일부를 제공하는 것
- 배포 목록은 제품 기능 목록의 항목 중에서 이번 배포 본에 포함하기로 결정한 것
- 배포 목록을 작성하면 이번 배포 본의 개발 범위와 일정을 수립할 수 있음
- 사용자에게 전달되는 배포 본의 기능 내역과 시기, 스프린트 주기, 배포 일정을 결정하게 됨

표 1-6 배포 목록

기능	내용	작업 일수(MD)	스프린트 주기
수강신청	학생이 수강신청을 할 수 있게 한다.	10MD	스프린트 1
성적등록	교수가 성적을 입력할 수 있게 한다.	7MD	스프린트 2
교과목, 개설과목등록	교수가 교과목과 개설과목을 등록할 수 있게 한다.	18MD	스프린트 3
총 MD	35MD		
총 배포 스프린트	3개		
배포 날짜		●	2021. 10. 31



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스크럼 개발 관련자의 역할

표 1-7 제품 책임자, 스크럼 마스터, 스크럼 팀의 역할

담당자	역할
제품 책임자	<ul style="list-style-type: none">제품 가능 목록을 만들비즈니스 관점에서 우선순위와 중요도를 매기고 새로운 항목을 추가함스프린트 계획 수립 시까지만 역할을 수행하고, 스프린트가 시작되면 팀 운영에 관여하지 않음
스크럼 마스터	<ul style="list-style-type: none">제품 책임자를 돋는 조력자업무를 배분만 하고, 일은 강요하지는 않음스크럼 팀이 스스로 조직하고 관리하도록 지원함개발 과정에서 스크럼의 원칙과 가치를 지키도록 지원함개발 과정에 방해될 만한 요소를 찾아 제거함
스크럼 팀	<ul style="list-style-type: none">팀원은 보통 5~9명으로 구성되며, 사용자 요구사항을 사용자 스토리로 도출하고 이를 구현함기능을 작업 단위로 나누고, 일정이나 속도를 추정해서 제품 책임자에게 알려줌하나의 스프린트에서 생산된 결과물을 제품 책임자에게 시연함매일 스크럼 회의에 참여하여 진척 상황을 점검함



08. 애자일 프로세스 모델



■ 스크럼 방식의 이해

▪ 스크럼 방식의 장점과 단점

• 스크럼 방식의 장점

- 반복 주기마다 생산되는 실행 가능한 제품을 통해 사용자와 충분히 의견을 나눌 수 있음
- 일일 회의를 함으로써 팀원들 간에 신속한 협조와 조율이 가능
- 일일 회의 시 직접 자신의 일정을 발표함으로써 업무에 집중할 수 있는 환경이 조성
- 다른 개발 방법론에 비해 단순하고 실천 지향적
- 스크럼 마스터는 개발 팀원들이 목표 달성을 집중할 수 있도록 팀의 문제를 해결
- 프로젝트의 진행 현황을 볼 수 있어 신속하게 목표와 결과 추정이 가능
- 프로젝트의 진행 현황을 볼 수 있어 목표에 맞게 변화를 시도할 수 있음

• 스크럼 방식의 단점

- ◉ 반복 주기가 끝날 때마다 실행 가능하거나 테스트할 수 있는 제품을 만들어야 하는데 이 작업이 많아지면 그만큼의 작업 시간이 더 필요
- 일일 스크럼 회의 시간(15분)이 넘어가면 작업시간이 늦어지고 작업하는데 방해 받을 수 있음
- 투입 공수를 측정하지 않기 때문에 작업이 얼마나 효율적으로 수행되었는지 알기 어려움
- 프로세스 품질을 평가하지 않기 때문에 품질 관련 활동이 미약하고 따라서 품질의 정 도를 알 수 없음



Time: 1.3

IT@COOKBOOK



쉽게 배우는 소프트웨어 공학

2판

Chapter 02 UML





목차

- 01 UML의 이해
- 02 유스케이스 다이어그램
- 03 클래스 다이어그램
- 04 순차 다이어그램
- 05 통신 다이어그램
- 06 활동 다이어그램
- 07 상태 다이어그램
- 08 컴포넌트 다이어그램
- 09 배치 다이어그램





학습목표

- UML 다이어그램을 이해한다.
- UML 다이어그램의 용도를 알아본다.
- 유스케이스 다이어그램을 자세히 알아본다.



01. UML의 이해



■ UML

▪ UML의 역할

- 소프트웨어의 전체를 판단할 수 있도록 12개의 다이어그램을 제시
- 시스템이 상호작용하는 측면, 시스템 전체 구조 측면, 컴포넌트 간의 관계 등을 시각적으로 볼 수 있게 나타낸 도면

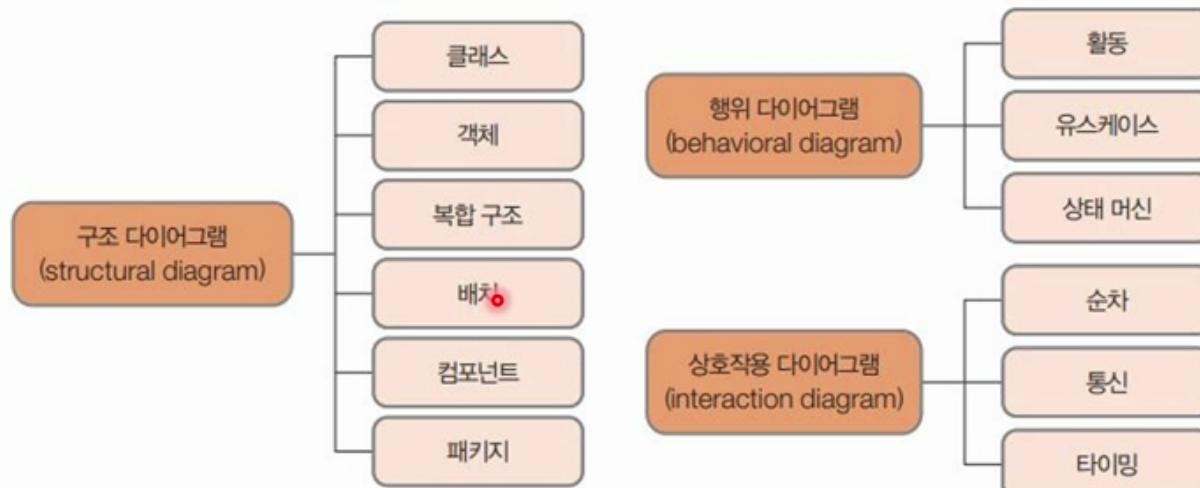


그림 2-1 UML의 12개 다이어그램



02. 유스케이스 다이어그램



■ 액터

▪ 액터의 종류

- 사용자 액터

- 사용자 액터는 시스템을 사용하는 사람(역할)을 의미



그림 2-2 액터의 표기

- 액터와 유스케이스의 관계는 화살표(→)를 사용해 표현하며 화살표 방향은 액터에서 유스케이스로 향함



그림 2-3 사용자 액터와 유스케이스의 관계



02. 유스케이스 디자인 프로그램



■ 액터

▪ 액터의 종류

- 시스템 액터

- 해당 프로젝트의 개발 범위에는 속하지 않지만 데이터를 주고받는 등 서로 연동되는 또 다른 시스템
- 유스케이스가 시스템 액터를 사용(참조)하는 것이므로 화살표 방향은 유스케이스에서 시스템 액터로 향함



그림 2-4 유스케이스와 시스템 액터의 관계

- 주요 액터

- 시스템에게 작업의 실행을 요구하는 능동적 입장의 액터
- 대부분의 액터가 여기에 해당

- 보조 액터

- 유스케이스로부터 요청을 받거나 메시지를 전달받아 수동적으로 작업을 하는 액터



02. 유스케이스 디아어그램



■ 액터

▪ 액터 식별

- 액터 식별 방법(학사관리시스템의 예)

- ① 개발 시스템을 사용하는 사람을 찾음

- 학사관리시스템을 사용하는 사람은 교수, 학생, 조교, 학사담당직원 등

- ② 시스템에 자료를 등록/수정/삭제/조회하는 사람을 찾음

- 학사관리시스템에서 액터는 성적을 등록/수정/삭제/조회하는 교수, 성적을 조회만 하는 학생, 조교, 학사담당직원

- ③ 개발 시스템에 연동된 또 다른 시스템을 찾음(시스템 액터)

- 앞에서 학사관리시스템에서 휴학신청 유스케이스는 도서관리 시스템

- ④ 시스템을 유지 및 관리하는 사람을 찾음(보조 액터)

- 시스템의 기능을 직접 사용하지는 않지만 시스템이 잘 구동되도록 유지 및 관리해주는 유지보수담당자

- ⑤ 유스케이스 기능에 권한은 없지만 역할을 대신하는 사람을 찾음(프록시 액터)



02. 유스케이스 디어그램



■ 액터

▪ 액터 이름

• 액터 이름 설정 규칙

- ① 이름만 봐도 액터의 의미를 알 수 있게 정함
- ② 역할을 나타내는 단어로 사용자 액터의 이름을 정함
 - 부장,과장 같은 직책이 아닌 역할로 정함
 - 구체적인 역할을 나타내는 단어를 사용
- ③ 시스템 액터는 시스템 이름을 사용해 이름을 정함



그림 2-6 역할을 나타내는 단어를 액터 이름에 사용한 예



02. 유스케이스 디자인



■ 액터

■ 액터 목록

- 각 액터에 관한 설명을 붙인 액터 목록을 만들면 그 액터가 시스템을 사용해 어떤 일을 하는지 이해하는 데 도움이 됨

표 2-1 액터 목록

이름	설명
교수	교수는 교과목과 개설과목을 결정하고 성적등록을 한다.
학생	학생은 수강신청, 성적조회, 휴학신청을 한다.
조교	조교는 교과목과 개설과목을 등록(대행)하고 휴학신청 승인을 한다. 조교는 성적장학생 명단을 등록한다.
학사담당직원	학사담당직원은 교과목과 개설과목의 등록 기간을 설정한다.



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스

- 사용자가 시스템을 통해 사용하고 싶은 기능
- 유스케이스가 모여 하나의 서브시스템을 이루고 서브시스템이 모여 개발 시스템이 됨
- 전체 시스템은 유스케이스를 모아 놓은 것과 같아야 함



그림 2-7 유스케이스 예

▪ 유스케이스 식별

- ① 사용자인 액터가 시스템에 요구하는 기능을 후보 유스케이스로 선정할 수 있음
 - ② 사용자의 업무에서 유스케이스를 도출할 수 있음
 - ③ 데이터베이스에서 데이터를 등록/수정/삭제/조회하는 기능을 하나의 유스케이스로 선정할 수 있음
- 찾은 후보 유스케이스는 여러 번의 정련 과정을 통해 최적화된 유스케이스로 결정
 - 유스케이스를 찾는 작업은 개발자가 하지만 사용자 관점에서 유스케이스를 정의해야 함



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 식별

- 처음에는 가능하다고 생각되는 것을 모두 도출해 놓고 하나씩 면밀히 살펴보는 과정을 거쳐 불필요한 것은 삭제하고, 필요시 합치거나 분리해 최종 유스케이스를 선정



그림 2-8 학사관리시스템에서 도출된 1차 후보 유스케이스



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 검증

- ① 시스템이 아니라 수작업으로 이루어지는 것은 유스케이스에 포함하면 안 됨
- ② 액터가 원하는 최종 결과만 나타내는 유스케이스인지 확인
- ③ 액터가 수행하는 유스케이스인지 확인
- ④ 유스케이스 내의 이벤트 흐름 전체를 액터가 사용하는지 확인



그림 2-9 수작업으로 이루어지는 것은 유스케이스 불가

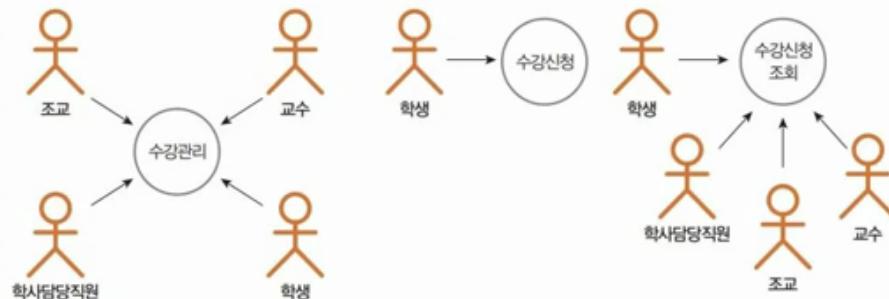


그림 2-10 유스케이스 검증



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 식별

- 처음에는 가능하다고 생각되는 것을 모두 도출해 놓고 하나씩 면밀히 살펴보는 과정을 거쳐 불필요한 것은 삭제하고, 필요시 합치거나 분리해 최종 유스케이스를 선정



그림 2-8 학사관리시스템에서 도출된 1차 후보 유스케이스



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 검증

- ① 시스템이 아니라 수작업으로 이루어지는 것은 유스케이스에 포함하면 안 됨
- ② 액터가 원하는 최종 결과만 나타내는 유스케이스인지 확인
- ③ 액터가 수행하는 유스케이스인지 확인
- ④ 유스케이스 내의 이벤트 흐름 전체를 액터가 사용하는지 확인



그림 2-9 수작업으로 이루어지는 것은 유스케이스 불가

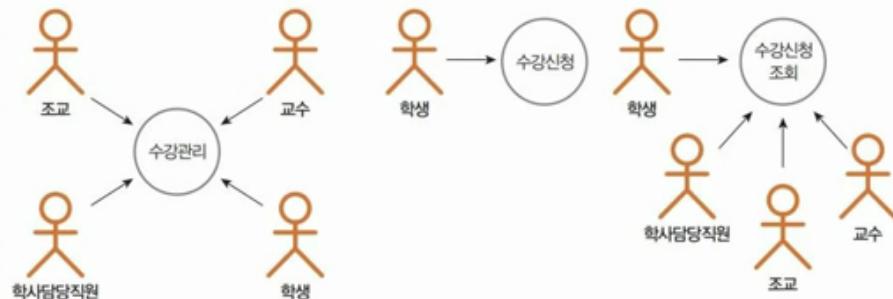


그림 2-10 유스케이스 검증



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 이름

- ① 사용자 관점에서 이해할 수 있는 이름을 사용



그림 2-11 사용자 관점의 유스케이스 이름

- ② 명사보다는 어떤 기능을 하는지 알 수 있는 동사형 명사를 사용



그림 2-12 동사형 명사를 사용한 이름



- ③ 사용자가 시스템을 통해 얻으려는 최종 목적이 나타나는 이름을 사용



그림 2-13 최종 목적을 사용한 이름



02. 유스케이스 디자인 프로그램



■ 유스케이스

▪ 유스케이스 목록

표 2-2 유스케이스 목록

이름	설명
교과목등록	교수는 교과목을 등록/수정/삭제한다.
개설과목등록	교수는 개설과목을 등록/수정/삭제한다.
성적입력	교수는 수강성적을 등록/수정/삭제한다.
강의계획서입력	교수는 강의계획서를 등록/수정/조회한다.
수강신청	학생은 수강과목을 등록/수정/삭제한다.
개인정보수정	학생은 개인정보를 수정/조회한다.
휴학신청	학생은 휴학을 신청한다.
휴학승인	조교는 학생이 신청한 휴학을 승인한다.
기간설정	학사담당직원은 교과목/개설과목/수강신청에 대한 기간을 설정한다.
출석부조회	교수/조교/학사담당직원은 출석부를 조회한다.
시간표/강의실조회	학생은 시간표/강의실을 조회한다.
성적조회	사용자(교수/학생/조교/학사담당직원)는 성적을 조회한다.
개설과목조회	사용자(교수/학생/조교/학사담당직원)는 개설과목을 조회한다.
교과목조회	사용자(교수/학생/조교/학사담당직원)는 교과목을 조회한다.
학생정보조회	사용자(교수/학생/조교/학사담당직원)는 학생정보를 조회한다.
강의계획서조회	사용자(교수/학생/조교/학사담당직원)는 강의계획서를 조회한다.



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 검증

- ① 시스템이 아니라 수작업으로 이루어지는 것은 유스케이스에 포함하면 안 됨
- ② 액터가 원하는 최종 결과만 나타내는 유스케이스인지 확인
- ③ 액터가 수행하는 유스케이스인지 확인
- ④ 유스케이스 내의 이벤트 흐름 전체를 액터가 사용하는지 확인



그림 2-9 수작업으로 이루어지는 것은 유스케이스 불가

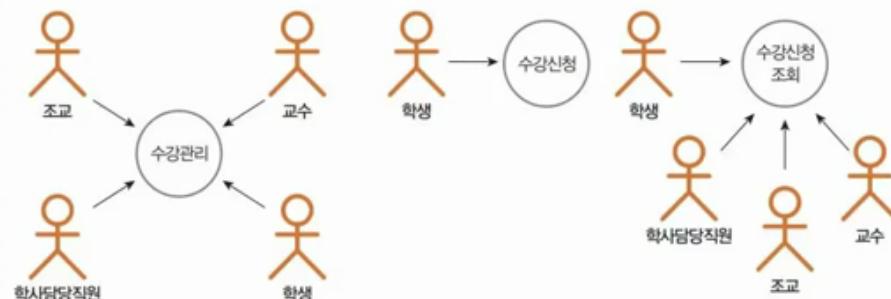


그림 2-10 유스케이스 검증



02. 유스케이스 디아어그램



■ 유스케이스

▪ 유스케이스 식별

- 처음에는 가능하다고 생각되는 것을 모두 도출해 놓고 하나씩 면밀히 살펴보는 과정을 거쳐 불필요한 것은 삭제하고, 필요시 합치거나 분리해 최종 유스케이스를 선정



그림 2-8 학사관리시스템에서 도출된 1차 후보 유스케이스



02. 유스케이스 디자인 프로그램



■ 유스케이스

▪ 유스케이스 목록

표 2-2 유스케이스 목록

이름	설명
교과목등록	교수는 교과목을 등록/수정/삭제한다.
개설과목등록	교수는 개설과목을 등록/수정/삭제한다.
성적입력	교수는 수강성적을 등록/수정/삭제한다.
강의계획서입력	교수는 강의계획서를 등록/수정/조회한다.
수강신청	학생은 수강과목을 등록/수정/삭제한다.
개인정보수정	학생은 개인정보를 수정/조회한다.
휴학신청	학생은 휴학을 신청한다.
휴학승인	조교는 학생이 신청한 휴학을 승인한다.
기간설정	학사담당직원은 교과목/개설과목/수강신청에 대한 기간을 설정한다.
출석부조회	교수/조교/학사담당직원은 출석부를 조회한다.
시간표/강의실조회	학생은 시간표/강의실을 조회한다.
성적조회	사용자(교수/학생/조교/학사담당직원)는 성적을 조회한다.
개설과목조회	사용자(교수/학생/조교/학사담당직원)는 개설과목을 조회한다.
교과목조회	사용자(교수/학생/조교/학사담당직원)는 교과목을 조회한다.
학생정보조회	사용자(교수/학생/조교/학사담당직원)는 학생정보를 조회한다.
강의계획서조회	사용자(교수/학생/조교/학사담당직원)는 강의계획서를 조회한다.



02. 유스케이스 디아어그램



■ 관계

■ 액터 → 유스케이스 관계

- 액터와 유스케이스는 연관 관계로 방향성을 갖는 실선으로 표현
- 화살표 방향은 데이터가 흘러가는 방향이 아니라 제어의 흐름을 나타냄
- 제어하는 주체에서 출발해 제어받는 대상으로 연관 방향이 결정



그림 2-14 액터와 유스케이스의 관계

- 시스템 액터도 유스케이스 제어의 주체가 될 수 있음
- 학생(액터)과 증명서자동발급기(시스템 액터) 사이인 액터와 액터 사이에도 연관 관계가 존재



그림 2-15 액터와 시스템 액터, 시스템 액터와 유스케이스 간의 연관 관계



02. 유스케이스 다이어그램



■ 관계

▪ 유스케이스 → 액터 관계

- 유스케이스의 수행 결과를 액터에게 알려줄 때는 유스케이스에서 액터로 방향성을 갖는 실선으로 표현
- 유스케이스 → 액터 관계에서 주의할 사항은 통보 기능이 시스템 내에서 이루어져야 함
- 유스케이스 다이어그램에서 화살표로 나타냈다면 그 기능을 개발에 포함해야 함

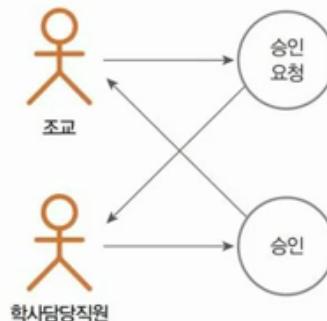


그림 2-16 유스케이스와 액터의 관계

• 유스케이스 → 시스템 액터 관계



그림 2-17 유스케이스와 시스템 액터의 관계

